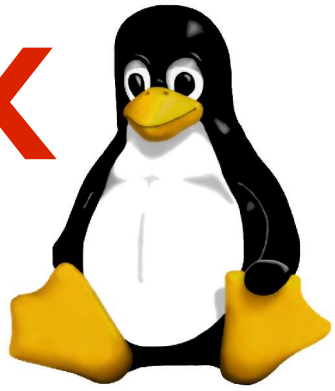
Linux Basic Lessons



book I

บทนำ

หากจะเอ่ยถึง Operating System (OS) หรือ "*ระบบปฏิบัติการ*" ทั้งหมดที่กำลังได้รับความสนใจจากผู้ใช้งานทั่วโลกในปัจจุบัน เราก็คงหลีกไม่พ้นที่จะต้องเอ่ยถึง Microsoft Windows, UNIX แล้วก็ ... โดยเฉพาะอย่างยิ่ง ... **Linux** ... OS ที่ถือว่าเป็น "*ญาติสนิท*" กับ UNIX มาแต่กำเนิด โดยยังคงรักษาสถานภาพความเป็น Open Source แบบ GPL (General Plublic License) หรือ software ที่มีลิขสิทธิ์เป็น "*ของสาธารณะ*" ตราบจนกระทั่งทุกวันนี้

อะไรคือ "ลิขสิทธิ์สาธารณะ"?

software ที่พัฒนาขึ้นมาภายใต้ GPL หรือ "*ลิขสิทธิ์สาธารณะ*" หมายถึง software ที่ "*ทุกคน*" สามารถที่จะเป็นเจ้าของ, จัดทำสำเนา, ดัดแปลง, แก้ไขเพิ่มเติม หรือจำหน่ายจ่ายแจกได้ "*อย่างเสรี*" โดย ไม่มีค่าใช้จ่ายใดๆ ที่เป็น "*ค่าลิขสิทธ์*" เข้ามาเกี่ยวข้องเลยตลอดทั้งกระบวนการ

ดังนั้นเอง ... ปัจจัยหนึ่งที่ทำให้ Linux กลายมาเป็น OS ยอดนิยมสำหรับนักคอมพิวเตอร์หลายต่อหลายคนทั่วโลก ก็เพราะ Linux เป็น OS ที่ให้ "*อิสรภาพ*" แก่นักพัฒนาโปรแกรมในการปรับแต่งทุกๆ อย่างตามที่แต่ละคนต้องการได้อย่างประหยัดที่สุด ... และถึงแม้ว่า Linux จะได้รับการพัฒนาต่อยอดมาจาก UNIX ซึ่งทำให้มีพื้นฐานการใช้งาน รวมไปถึงโครงสร้างและรูปแบบของคำสั่งที่ใกล้เคียงกับ UNIX มากจนเกือบจะนับให้เป็น OS ในตระกูลเดียวกันได้ ... ***แต่ Linux ก็ไม่ใช่ UNIX อยู่ดี*** ... เพราะส่วนที่เป็น source code ในระดับ kernel ของ Linux นั้น ถูกพัฒนาขึ้นมาจากการผสมผสาน source code ของกลุ่มนักพัฒนาโปรแกรมคอมพิวเตอร์อิสระที่เรียกตัวเองว่า GNU (ย่อมาจากชื่อเต็มๆ ว่า Gnu's Not UNIX เป็นการเล่นคำของชื่อเต็มกับชื่อย่อให้สะกดวนทับซ้อนกัน) ซึ่งเป็น source code ที่เขียนขึ้นมาภายใต้เงื่อนไขของ GPL หรือ "*ลิขสิทธ์สาธารณะ*" เช่นเดียวกัน ... จึงทำให้หลายๆ คนคิดว่า Linux ควรจะมีชื่อเรียกเต็มๆ ว่า GNU Linux ด้วย ... แต่นั่นก็ไม่ใช่ข้อบังคับของ GNU อยู่ดี !!

จุดกำเนิดที่แท้จริงของ Linux นั้นถือกำเนิดมาจากความต้องการที่จะจำลองระบบปฏิบัติการ UNIX ให้สามารถใช้งานได้บนเครื่องคอมพิวเตอร์ส่วนบุคคล เพื่อช่วยให้นักศึกษาทุกคนสามารถเรียนรู้และฝึกหัดการใช้งาน UNIX ได้บนเครื่องคอมพิวเตอร์ที่มีระดับราคาไม่สูงมากจนเกินไป (UNIX เป็น OS ที่มีลิขสิทธิ์เชิงพาณิชย์ และมีการจัดจำหน่ายในระดับราคาที่เหมาะกับองค์กรขนาดใหญ่เท่านั้น ทั้งยังถูกพัฒนาขึ้นมาให้ทำงานบนเครื่องระดับ Mini Computer และ Mainframe ซึ่งแพงเกินกว่าที่จะมีไว้ในกรรมสิทธิ์ระดับบุคคลทั่วๆ ไปได้) ... ในเวลานั้นก็มี software เล็กๆ ตัวหนึ่งที่ชื่อว่า Minix ถูกพัฒนาขึ้นมาโดย Andy Tanenbaum และแจกจ่ายให้ฟรีแก่นักศึกษาทั่วไปเพื่อจุดประสงค์ดังกล่าวอยู่แล้ว ... อย่างไรก็ตาม Minix ถูกกำหนดขอบเขตให้อยู่เพียงในกรอบของวัตถุประสงค์ทางการศึกษา จึงไม่ได้รับความสนใจที่จะพัฒนาต่อยอดขึ้นไปให้เป็น OS อย่างสมบูรณ์ด้วยตัวของมันเอง ... จนกระทั่งนักศึกษาแห่งมหาวิทยาลัย Helsinki คนหนึ่ง ชื่อว่า Linus Torvalds คันไม้คันมือจนกระโดดเข้ามาจัดการเรื่องนี้ด้วยตัวเองเมื่อปี 1991 ... และนั่นก็คือปีเกิดของ Linux

Linux เป็น OS ที่ได้รับการพัฒนาอย่างต่อเนื่องนับตั้งแต่วันแรกๆ ของมัน โดยอาศัยความร่วมมือจากนักพัฒนาโปรแกรมคอมพิวเตอร์อิสระทั่วโลก มีการสื่อสารข้อมูลเพื่อแลกเปลี่ยนความคิดและความรู้ รวมไปถึง source code ระหว่างกันผ่านเครือข่าย internet ... เป็น "*ผลิตผลของชุมชนขนาดใหญ่*" ในโลก Cyberspace ... จึงเป็นเรื่องธรรมดาสำหรับนักคอมพิวเตอร์ที่มีส่วนร่วมในเรื่องนี้ทุกๆ คน ที่จะรู้สึกถึง "*ความเป็นเจ้าของร่วมกัน*" ของพวกเขาตลอดมา

เป็นเวลากว่า 10 ปีที่ Linux ได้รับการร่วมพัฒนาโดยนักคอมพิวเตอร์นับล้านคนทั่วโลก ... จนกระทั่งทุกวันนี้ ... Linux ได้รับการพิสูจน์แล้วว่า มันคือหนึ่งใน OS ที่มีประสิทธิภาพและมีเสถียรภาพสูงมาก มีการใช้งานเป็น "*เครื่องแม่ข่าย*" หรือ server จริงบน internet เกินกว่า 60% ของทั้งหมด และมีการใช้งานภายในองค์กรธุรกิจทั้งขนาดใหญ่และขนาดเล็กอีกจำนวนนับไม่ถ้วน ถือเป็นระบบปฏิบัติการหรือ OS ที่มีผลกระทบต่อธุรกิจในวงการอุตสาหกรรมคอมพิวเตอร์อย่างมหาศาล ... เพราะ Linux ยังเป็น Open Source แบบ GPL ที่ไม่ได้เรียกเก็บ "*ค่าลิขสิทธิ์*" ใดๆ จากผู้ใช้งานเลย ... เนื่องจากลิขสิทธิ์นั้นยังเป็นของ "*สาธารณชน*" ทุกๆ คน

อย่างไรก็ตาม เราต้องไม่ลืมว่า Linux นั้นถูกพัฒนาโดย "*นักคอมพิวเตอร์อิสระ*" ซึ่งส่วนใหญ่ก็จะมีความเชี่ยวชาญและช่ำชองกับการใช้งานคอมพิวเตอร์อยู่ในระดับที่สูงถึงสูงมาก ... ทั้งยังเริ่มต้นการพัฒนาโดยให้มีโครงสร้างของคำสั่งใช้งานแบบ UNIX ซึ่งมีประวัติศาสตร์ยาวนานกว่า 40 ปีอีกด้วย ... ดังนั้น Linux ที่พวกเราเห็นในทุกวันนี้ จึงเป็นผลผลิตที่เกิดจากการสะสมของประสบการณ์ที่ไม่ใช่ระดับของ user ปกติทั่วไป ... ทักษะและมุมมองที่ผู้ใช้งานจะต้องเข้าไปเกี่ยวข้อง จึงแตกต่างจาก OS อย่าง Microsoft Windows หรือ MacOS ที่เน้นการใช้งานในระดับของ user มาตั้งแต่แรก ... แต่ Linux ถูกเน้นหนักไปในเรื่องของความยืดหยุ่น, ความปลอดภัย, ความมีประสิทธิภาพ, ความมีเสถียรภาพ ซึ่งจำเป็นต้องอาศัยความเป็นระบบระเบียบของการจัดการ เพื่อจะสามารถตอบสนองความต้องการในระดับเครือข่าย computer network มากกว่าจะเป็นการใช้งานแบบ stand alone ที่ไม่ยอมเกี่ยวข้องกับใคร

โดยเหตุนี้เอง การเรียนรู้และศึกษา Linux จึงจำเป็นที่จะต้องทำอย่างมีระบบระเบียบพอสมควร เพื่อที่จะจัดกลุ่มก้อนของประสบการณ์อันหลากหลายที่สะสมรวมกันอยู่ใน Linux ให้สามารถถูกมองเห็นเป็นส่วนๆ ได้อย่างชัดเจนมากขึ้น ... ถึงแม้จะมีคำกล่าวอ้างที่ว่า "*ในระดับการใช้งานของ user ปกตินั้น ไม่จำเป็นต้องเรียนรู้รายละเอียดในทุกๆ เรื่อง*" ก็ตาม แต่การแจกแจงรายละเอียดที่สะสมอยู่ใน Linux ตลอดช่วงเวลากว่า 10 ปีที่ผ่านมานั้น จะช่วยเป็นสื่อให้ user ทั่วไปสามารถเข้าใจได้ว่าส่วนไหนของ Linux บ้างที่เขาไม่จำเป็นต้องเรียนรู้เลย หรือมีส่วนไหนที่อยู่ในอำนาจควบคุมที่ user สามารถจัดการได้ด้วยตัวเองอย่างมีประสิทธิภาพ

เป็นที่น่าเสียดายที่นักคอมพิวเตอร์เก่งๆ ในโลกของ Linux จะเก่งเฉพาะในเรื่องของการนำความรู้มาใช้เพื่อพัฒนาโปรแกรม หรือไม่ก็แค่เก่งในการบริหารจัดการระบบเท่านั้น แต่ไม่ได้เก่งมากนักกับการถ่ายทอดความรู้ให้กับ user ทั่วๆ ไป เพื่อพัฒนาคนที่ยังจะต้องมีส่วนเกี่ยวข้องอยู่ในระบบด้วยเสมอ ... การเขียนตำราว่าด้วย Linux ฉบับนี้จึงถูกเขียนขึ้นมาจากมุมมองของ user คนหนึ่ง โดยหวังที่จะถ่ายทอดประสบการณ์ขั้นพื้นฐานนี้ให้กับ user อื่นๆ ด้วยการพยายามคัดสรรแยกแยะกลุ่มก้อนของประสบการณ์ที่รวบรวมอยู่ใน Linux นั้นออกมาเป็นหมวดหมู่ที่ชัดเจน เพื่อให้มองเห็นภาพรวมของ OS ที่ทรงประสิทธิภาพตัวนี้อย่างเต็มที่ ก่อนที่จะเจาะลึกลงไปศึกษารายละเอียดของแต่ละส่วนอย่างจริงๆ จังๆ ต่อไป.

Mr.Z.
กรุงเทพฯ, 09.11.2004

Studying Outline of **Linux**

Partitioning & Installation

- **/** — root file system
  - /bin : binary files / command
  - /dev : device drivers
  - /etc : configuration file
  - /lib : kernel modules
  - /sbin : superuser binary files /cmd.
- **/boot** : boot images
- **/usr** : system files & software
- **/var** : log file & web/mail server
- **/tmp** : temporary files
- **/swap** : swap file system
- **/home** : user's files

Bootup Sequence

ขั้นตอนในการ boot / ลำดับของการเข้าสู่ระบบ

System Admin

- **User & Group Account Management**
- **File System**
  - RAIDS
  - LVM
  - NFS
  - SMBFS (Samba)
  - Local
- **Services**
  - DNS
  - httpd / Apache
  - squid / proxy
  - e-mail
- **Securities**
  - PAM
  - NIS
  - iptables / firewalls
  - tcp / wrappers

โครงร่างของบทเรียน

ผมจัดการแบ่งหัวข้อของ "*แบบเรียน*" Linux ออกเป็น 3 กลุ่มใหญ่ๆ คือ

1. Partitioning & Installation : เป็นรายละเอียดของการติดตั้งระบบ
2. Bootup Sequence : เป็นรายละเอียดของขั้นตอนในการ boot / ลำดับของการเข้าสู่ระบบ
3. System Administration : เป็นเรื่องของการดูแลรักษา, การปรับแต่ง, หรือติดตั้งเพิ่มเติม

โดยในแต่ละกลุ่มที่จัดแบ่งไว้นี้ ก็จะมีรายละเอียดปลีกย่อยออกไปอีกหลายเรื่องด้วยกัน ซึ่งผมทำตารางสรุปคร่าวๆ ไว้ในหน้าที่ 3 แล้ว ... ผมสร้างตารางนี้ขึ้นมาด้วย concept แบบ mind map เพื่อใช้เป็น "*แผนที่*" หรือ "*ตุ๊กตา*" ที่จะเล่ารายละเอียดต่อไปทีละเรื่องๆ จนกว่าจะจบเอกสารทั้งฉบับ ... ในขณะเดียวกัน ผมก็หวังว่า "*แผนที่*" ที่เห็นเป็นตารางที่ว่านี้ น่าจะมีส่วนช่วยให้ผู้ที่เริ่มต้นศึกษา Linux สามารถมองเห็นหมวดหมู่ของระบบต่างๆ ที่ทับซ้อนกันเป็นโครงสร้างหลักๆ ของ OS ตัวนี้อย่างชัดเจนมากขึ้นด้วย

อย่างไรก็ตาม ผมจะเริ่มต้นเล่าจากกลุ่มที่ 1 ไปกลุ่มที่ 2 แล้วก็จะแทรกเรื่องราวของคำสั่งและการใช้งานพื้นๆ ทั่วๆ ไป ก่อนที่จะเลยเถิดไปถึงรายละเอียดในระดับของ System Admin ... ซึ่งก็เป็นไปได้ว่าผมจะต้องแยกเอกสารนี้ออกเป็น 2 ฉบับ โดยเอาส่วนของ System Admin แยกไปเล่าต่างหากในแบบที่ตัวเองต้องค้นคว้าเพิ่มเติม ซึ่งก็จะถือโอกาสเรียนรู้ในเชิงลึกเกี่ยวกับ Linux ในระหว่างการเรียบเรียงเอกสารท่อนสุดท้ายนั้นด้วยเหมือนกัน

ก่อนที่เริ่มต้นเข้าสู่เนื้อหาของเอกสารฉบับนี้ ผมขอออกตัวไว้ก่อนว่า นี่คือตำราที่เขียนจากมุมมองของ user ธรรมดาๆ คนหนึ่ง ถือเป็นการเล่าสู่กันฟังมากกว่าการบรรยาย แล้วก็ไม่ค่อยจะเน้นในเรื่องของความแม่นยำถูกต้อง 100% ... แต่เอาแค่ยังสามารถมีชีวิตรอดต่อไปได้กับเครื่องคอมพิวเตอร์ส่วนบุคคลที่เห็นตำหน้าตำตากันอยู่ทุกวี่ทุกวัน ... ก็พยายามอย่างที่สุดครับที่จะให้เนื้อหานั้นตรงไปตรงมาอย่างเข้าใจได้ง่ายๆ ... บางท่อนบางตอนอาจจะเขียนรายละเอียดจน 'เวอร์ไปหน่อย แต่เกือบทั้งหมดผมก็เพิ่งจะเรียนรู้เป็นครั้งแรกเหมือนกัน ... ทั้งคนเขียนกับคนอ่านไม่ได้มีความรู้ดีไปกว่ากันมากนักหรอกครับ ... พวกเราล้วนแล้วแต่เริ่มต้นกันที่ "ศูนย์" ทั้งนั้นแหละคราวนี้ :-) ... ผมยืนยันได้แค่ว่าการได้เรียนรู้สิ่งที่เราไม่เคยรับรู้มาก่อนเลยนั้น เป็นความสุขและความสนุกอย่างที่สุดเสมอ !!

ส่วนที่ 1 : Partitioning & Installation

เราจะเริ่มต้นกันตั้งแต่การติดตั้งระบบ ... ?!

แม้ว่า user ส่วนใหญ่จะมองว่าการติดตั้งระบบเป็นเรื่องทางเทคนิคที่ "*ผู้ใช้งานทั่วไป*" ไม่น่าจะต้องเข้าไปเกี่ยวข้อง แต่ก็อย่างที่บอกไว้ตั้งแต่แรกแล้วว่า การเรียนรู้เกี่ยวกับ Linux นั้นจำเป็นที่จะต้องปรับทัศนคติบางอย่างของ user ซะก่อนเป็นอันดับแรก ... โดยเฉพาะในเรื่องของการดูแล "*เครื่องมือ*" ของตัวเอง

ทุกวันนี้ Linux ที่มีการจำหน่ายจ่ายแจกกันในตลาด จะมีอยู่ด้วยกันหลายค่ายครับ และทุก ๆ ค่ายก็พยายามที่จะอำนวยความสะดวกให้กับ user มากขึ้นกว่ายุคแรก ๆ ของมันแล้วด้วย อย่างเช่น Red Hat, Mandrake, SuSE รวมไปถึง YellowDog หรือของไทยๆ อย่าง SiS, Grand Linux, Linux TLE, ฯลฯ (ซึ่งสองตัวหลังนี่ไปเอา Red Hat มาใส่ภาษาไทยครับ เช่นเดียวกับ Mandrake ที่เริ่มต้นจากการนำ Linux ของ Red Hat ไปดัดแปลงมาก่อน ส่วน YellowDog ก็เอา Linux ของ Red Hat ไปดัดแปลงให้ใช้งานได้กับเครื่อง Macintosh หรือเครื่องคอมพิวเตอร์ที่ใช้ CPU แบบ PowerPC) โดยส่วนใหญ่ก็จะพยายาม "*แข่งกันง่าย*" ในปัจจุบัน ยกเว้น Linux "*หัวเอียงซ้าย*" อย่าง Slackware ที่ยังคงความเป็นอมตะนิรันดร์กาลของเหล่านักพัฒนาโปรแกรมคอมพิวเตอร์รุ่นแรกๆ ของ Linux ตราบจนกระทั่งปัจจุบัน ... ค่ายต่างๆ ของ Linux เริ่มที่จะรวมตัวกันบ้างและเริ่มที่จะถูก take over โดยยักษ์ใหญ่ในวงการ software บ้างแล้ว แต่ก็เกิดมีค่ายใหม่ๆ เข้ามาแซมอยู่เรื่อยๆ เพราะการดัดต่อเพิ่มเติมหรือดัดแปลง Linux นั้นเป็น "*ความอิสระ*" ที่ใครๆ ก็สามารถทำได้อยู่แล้ว ... **สำหรับในปัจจุบัน ค่ายที่ค่อนข้างจะมั่นคงและมีอิทธิพลมากที่สุดก็น่าจะเป็นค่ายของ Red Hat เพราะตั้งใจที่จะทำตัวเป็นองค์กรทางธุรกิจมาตั้งแต่แรก** ต่างจาก Linux ค่ายอื่นๆ ที่ตั้งใจกิน "*อุดมการณ์*" ต่างอาหารหลัก และจบลงด้วยการ "ขายชื่อ" หรือ Brand Name ของตัวเองให้กับยักษ์ใหญ่เขาไปบริหารเพื่อการบริโภคกันต่อไป

Linux แจกจ่ายได้ฟรี ๆ แต่ไม่ได้แปลว่าห้ามทำมาค้าขาย !! เพราะ Linux ตัวจริง ๆ นั้นมันเป็นแค่ kernel ที่เป็นหัวใจสำคัญของทั้งระบบเท่านั้นเอง ... แต่การที่เราเห็น Linux มีอยู่ด้วยกันหลายค่ายหลายสำนักนั้น ล้วนแล้วแต่เป็นสิ่งที่แต่งเติมเข้าไปให้กับ kernel หรือ Linux แท้ ๆ เพื่ออำนวยความสะดวกให้กับ user แต่ละกลุ่มแต่ละพวกอย่างเฉพาะเจาะจง ... อย่างเช่น Slackware จะเน้นไปที่กลุ่มของ System Admin ที่มีความรู้ความชำนาญในการติดตั้งและดูแลระบบเครือข่าย ... ในขณะที่ SuSE จะเน้นที่มี software แถมฟรีมาให้มากมายเพื่อให้ user เลือกใช้ได้ตามใจชอบ ... หรือ Mandrake ที่หลายคนมักจะบอกว่าเหมาะกับการทำเป็น Workstation แต่ไม่เหมาะกับงาน Server ... ส่วนค่ายของ Red Hat ก็ประกาศตัวผลิตภัณฑ์ของตัวเองออกมาเป็น Workstation บ้าง เป็น Enterprise Server บ้าง หรือเป็น Advanced Server บ้าง แตกต่างกันไปหลายระดับองค์กรด้วยกัน ยังไม่นับรวมค่ายอื่นๆ อีกหลายค่าย ... **เป็น *"อิสรภาพที่น่าปวดหัว"* พอสมควรทีเดียว**

การเลือกใช้ Linux ค่ายไหนไม่สำคัญเท่ากับที่เราจะต้องเข้าใจว่า kernel หรือตัวแกนกลางของระบบจริง ๆ นั้นมันคือตัวเดียวกันเสมอ ... *นี่ถึงจะเป็นจุดเริ่มต้นของการใช้งานและการเรียนรู้* ... ระบบปฏิบัติการหรือ OS ตัวอื่นๆ อย่าง Microsoft Windows มักจะมีการ preloaded อยู่ในเครื่องคอมพิวเตอร์หลายยี่ห้ออยู่แล้ว ... หรือ MacOS ก็ทำการ preloaded มาจากโรงงานซะเลย

เพราะเขาเป็นเจ้าของลิขสิทธิ์ทั้ง hardware และ software มาตั้งแต่ไหนแต่ไร ... ส่วน *Linux มักจะมาถึงเราในสภาพที่เราต้องจัดการติดตั้งด้วยตัวเองเท่านั้น คนที่อยากจะใช้งาน Linux จึงต้องเป็นมนุษย์ประเภทที่ "มือคัน" พอสมควร*

มีแง่มุมเล็กๆ อยู่เรื่องหนึ่งที่อยากจะบอกเอาไว้ ... ส่วนใหญ่ของผู้ที่ใช้งาน Linux นั้นมักจะเป็นพวก "*ผู้ต่อต้าน*" software เถื่อน หรือของที่ละเมิดลิขสิทธิ์ พวกเขาจะให้ความเคารพในทรัพย์สินทางปัญญาของคนอื่นๆ ค่อนข้างมาก ไม่ว่าเจ้าของลิขสิทธิ์นั้นจะได้รับความจงเกลียดจงชังจากชาวโลกขนาดไหนก็ตาม ... software ฟรีในความหมายของพวกเขา คือ software ที่เจ้าของลิขสิทธิ์ "*ยินยอม*" ให้ใช้งานกันฟรีๆ เท่านั้น โดยไม่นับรวม software เถื่อนที่ลักลอบขโมยรหัสผ่านมาใช้งานกัน ทั้งๆ ที่เจ้าของลิขสิทธิ์ไม่ยินยอมพร้อมใจ ... เรียกว่า "*ไม่ให้ก็ไม่ง้อ*" เพราะพวกเขารวมตัวกันพัฒนา software ขึ้นมาให้ใช้งานกันฟรีๆ อย่างถูกต้องตามกฎหมายอยู่ตลอดเวลาอยู่แล้ว ... เป็นกลุ่มบุคคลที่มี "*ความมือคัน*" อย่างมีศักดิ์ศรีที่ถูกที่ควรตามกฎหมายครับ

ดังนั้น หากมีใครย้อนถามขึ้นมาว่า ในเมื่อเรามี MS Windows ที่ใช้งานกันง่ายๆ อยู่แล้ว ยังจะขยันหาตำแยมาใส่มือเพื่ออะไรกัน?" ... เหตุผลก็คือการที่เรามี MS Windows ไว้ใช้งานกันง่ายๆ นั้น ส่วนใหญ่มันไม่ถูกกฎหมายไงล่ะครับ!! ... *ซึ่งบางครั้งผมเองก็ไม่แน่ใจเหมือนกันว่า การที่ชาวโลกตั้งแง่รังเกียจนาย Bill Gates และ Microsoft นั้น เพราะเขาน่าเกลียดจริงๆ หรือเพราะพวกเราอยากจะมีข้ออ้างที่ไม่ยอมจ่ายค่าลิขสิทธิ์กันแน่?!* ... ผมเคยอ่านทั้งประวัติและข้อเขียนในหนังสือของ Bill Gates มาก่อนแล้ว แต่ก็ยังไม่เห็นแง่มุมไหนที่ควรจะถูกจงเกลียดจงชังกันขนาดนั้น ยกเว้นอุปนิสัยส่วนตัวที่ค่อนข้างจะก้าวร้าว และไม่แคร์ความรู้สึกแบบมนุษย์ด้วยกันเท่านั้นเอง ... แต่ว่าคนทั้งโลกมีโอกาสได้สัมผัสกับประสบการณ์ทางตรงอย่างนั้นซักกี่คน? พวกเราเกือบทั้งหมดรับรู้เรื่องราวของเขาผ่านสื่อ ผ่านข้อเขียน หรือผ่านคำบอกเล่าต่อๆ กันมา ... แล้วก็รุมกันเกลียดเขา?! ... หรือเพราะพวกเราไม่อยากจ่ายค่าลิขสิทธิ์ให้กับเขา พวกเราถึงต้องแกล้งทำเป็นเกลียดกันจริงๆ?! ... จะว่าไปแล้ว ผมเองก็รู้สึกนิยมชมชอบคนอย่าง Bill Gates อยู่พอสมควร ... แต่ว่าผมรักวิธีคิดแบบของ Linus Torvalds มากกว่า ...

แต่ไม่ว่าเราจะรู้สึกรักใครหรือเกลียดใคร หากเรา "*มือคัน*" พอที่จะติดตั้ง OS ตัวใดตัวหนึ่งด้วยตัวเองแล้วล่ะก้อ ... ยากง่ายต่างกันไม่มากนักหรอกครับ ... แล้วที่สำคัญก็คือ หากเราคิดที่จะติดตั้งแบบไม่สนใจเรียนรู้อะไรเลยนะ ... Linux จากค่ายใหญ่ๆ อย่าง Red Hat, SuSE, Mandrake จะติดตั้งง่ายกว่าเดิมมากแล้วครับในเวลานี้!! ... แล้วถ้าหากว่าเรายัง "*มือคัน*" พอที่จะซุกซนทางปัญญาอีก การติดตั้ง Linux นั้นก็จะมีเรื่องราวที่เปิดโอกาสให้เราได้เรียนรู้อีกมากมายกว่า OS ตัวอื่นๆ ทั้งหมดด้วย ... เพราะนี่คือโลกของ Open Source !!

เอารึยัง? ... เราจะเริ่มกันตั้งแต่ ... *การแบ่ง Partitions* !!

เรื่องเล่าเกี่ยวกับ Partitions

คำว่า Partitions แปลตรงๆ ตัวว่า "*ฉากกั้น*" เมื่อถูกนำมาใช้ในโลกของ hard disk ที่ติดตั้งอยู่ในเครื่องคอมพิวเตอร์ทั่วๆ ไปนั้นจึงหมายถึง "*สิ่งที่แบ่งพื้นที่ของ hard disk ออกเป็นส่วนๆ*" ... คือแทนที่จะปล่อยให้เครื่องคอมพิวเตอร์เห็นพื้นที่ทั้งหมดของ hard disk เป็นเนื้อเดียวต่อเนื่องกันโดยตลอด เราก็จัดการสั่งให้คอมพิวเตอร์มอง hard disk นั้นๆ แยกออกเป็นท่อนๆ โดยมีจุดเริ่มต้นและจุดสิ้นสุดที่แน่นอนลงไป ... คล้ายๆ กับการนำชั้นวางของมาแบ่งซอยช่องว่างๆ ในตู้เอกสารออกเป็นชั้นๆ ช่องๆ อย่างนั้นแหละ

หลายคนน่าจะสงสัยว่าทำไปเพื่ออะไร? ในเมื่อเราก็สามารถสร้าง folders หรือ directories แยก

ต่างหากกันเพื่อจัดเก็บไฟล์ที่มีหลากหลายเรื่องราวและประเภทของมันได้อย่างเรียบร้อยอยู่แล้ว ยังจะต้องไปยุ่งเกี่ยวกับเรื่องของ partitions อีกทำไม?

โดยปกติแล้ว การแบ่ง folders หรือ directories นั้นเป็นเรื่องของการจัดเก็บ *"ไฟล์ข้อมูล"* ซึ่งเป็นเรื่องพื้นฐานที่ user ควรจะต้องเข้าใจไว้เป็น routine ของชีวิตการทำงาน (พวกที่จัดการกับเรื่องนี้ไม่ได้แปลว่าไม่เข้าใจชีวิต ... ว่างั้น!!) ... แต่การจัดแบ่ง partitions เป็นเรื่องของระบบครับ !!

เอางี้นะ ...

... ลองนึกถึงภาพห้องทำงานในสำนักงานทั่วๆ ไปก็แล้วกัน ... ถึงแม้ว่าเราจะมีแฟ้มใส่เอกสารแบ่งเป็นพวกๆ .. แยกเป็นเรื่องๆ .. แล้วในแต่ละแฟ้มก็ยังจัดเรียงเป็นหมวดหมู่ที่ชัดเจนด้วยแถบชื่อหรืออุปกรณ์อะไรอย่างอื่นก็ตาม ... มันก็เรียบร้อยดีใช่มั้ยล่ะ? ... แต่เราก็ไม่วางมันไว้ตามพื้น หรือกองสุมไว้บนโต๊ะทำงานตลอดเวลา ... ใช่รึเปล่า?

... เอาเรื่องพื้นที่ใช้สอยในสำนักงานอีกก็แล้วกัน การที่เราไม่ได้ใช้พื้นที่รวมๆ กันทุกอย่างในห้องโล่งๆ แต่ต้องแบ่งเป็นห้องทำงาน, ห้องรับแขก, ห้องประชุม, ห้องน้ำฯ, หรือห้องเก็บเอกสาร ฯลฯ เพราะมันเป็นเรื่องของความเป็นระบบระเบียบ และความเป็นสัดส่วนของพื้นที่ใช้สอยที่ชัดเจน ... ใช่รึเปล่า?

การจัดแบ่ง partitions ให้กับ hard disk ของเครื่องคอมพิวเตอร์ ก็เพื่อความเป็นระบบระเบียบของการใช้สอยพื้นที่ทั้งหมดใน hard disk นั่นเอง ... นี่ ... concept ของมันก็คืออยู่ตรงนี้ !!

แล้วยังไงถึงเรียกว่าเป็นระบบระเบียบล่ะ?

โดยปกติแล้ว เราจะแบ่งพื้นที่ส่วนที่เป็น OS ออกไปต่างหากเลยท่อนหนึ่ง แล้วก็แยกส่วนที่เก็บ application software หรือโปรแกรมประยุกต์สำหรับใช้งานอื่นๆ ออกไปอีกท่อนหนึ่ง จากนั้นก็เป็นพื้นที่ส่วนที่เราเก็บไฟล์ข้อมูลหรืองานต่างๆ ของเรา ... ซึ่งโดยทฤษฎีแล้วพื้นที่ทั้ง 3 ส่วนนี้ไม่ควรจะปะปนกัน เพราะ software ส่วนที่เป็น OS อาจจะต้องมีการ upgrade หรือไม่ก็ต้องติดตั้งซ่อมแซมในกรณีที่เสียหาย หรือติดตั้งส่วนประกอบอื่นๆ เพิ่มเติม ... ซึ่งไม่แน่ว่าเราอาจจะโชคร้ายที่ต้องล้างไส้ล้างพุงทิ้งทั้งหมดด้วยรึเปล่า ... ดังนั้นจึงไม่ควรเสี่ยงให้มันปะปนกับพื้นที่ที่เราใช้จัดเก็บไฟล์ข้อมูล !! ... ในขณะที่ OS มักจะมีขนาดคงที่ ไม่ค่อยจะมีการเปลี่ยนแปลงที่หวือหวามากนัก software ประเภท application กลับมีโอกาสที่จะเพิ่มเติมจำนวนเข้าไปได้เรื่อยๆ ขึ้นอยู่กับความหลากหลายของงานที่เราจำเป็นต้องใช้ ... ดังนั้น เขาจึงมักจะแยกพื้นที่ส่วนหนึ่งให้เอาไว้เก็บ application โดยเฉพาะ เพื่อที่มันจะไม่รุกล้ำไปกินพื้นที่ส่วนอื่นๆ ของ hard disk จนเต็ม ... เพราะถึงแม้ว่าจะไม่มีอะไรเสียหายเลยในระบบ แต่หากพื้นที่ของ hard disk เกิดเต็มขึ้นมาจริงๆ ... ระบบนั้นๆ ก็จะไม่สามารถ boot ขึ้นมาได้อีกเหมือนกัน ... **เป็นอันตรายที่หลายคนไม่ทันจะคำนึงถึงครับเรื่องนี้ !!**

พื้นที่ที่เราใช้เก็บข้อมูลก็เหมือนกัน เป็นอีกพื้นที่หนึ่งที่มีแต่การใช้มากขึ้นเรื่อยๆ ตลอดเวลา และน่าจะมีอัตราการเพิ่มของพื้นที่มากกว่าส่วนที่เป็น application ซะอีก ... ในขณะเดียวกัน การเปลี่ยนแปลงของข้อมูลในพื้นที่ตรงนี้ก็แทบจะมีการเปลี่ยนแปลงเป็นรายวัน ซึ่งบ่อยครั้งที่ user มักจะต้องการ backup หรือทำสำเนาข้อมูลแยกออกไปเก็บต่างหาก ... และไม่มีเหตุผลที่จะปล่อยให้ข้อมูลที่มักจะเปลี่ยนแปลงไปปะปนกับข้อมูลส่วนที่ไม่ค่อยจะมีการเปลี่ยนแปลงอย่าง application software ... โดยเหตุนี้ พื้นที่ส่วนที่ใช้เพื่อการจัดเก็บไฟล์ข้อมูล จึงมักจะถูกแยกออกไปต่างหาก ทั้งเพื่อเหตุผลของการ backup และเพื่อความปลอดภัยในกรณีที่ระบบเกิดล่มขึ้นมาจนต้องติดตั้งใหม่ ... เพราะอย่างน้อยที่สุด hard disk ส่วนที่แบ่งไว้เพื่อเก็บข้อมูลนี้ ก็จะไม่ต้องเข้าไปพัวพันกับการล้างไส้ล้างพุงครั้งไหนๆ ด้วยเลย

เป็นที่น่าเสียดายที่ระบบปฏิบัติการทั่วไปไม่พยายามพูดถึงเรื่องนี้ให้ชัดเจน โดยเฉพาะ Microsoft

Windows และ MacOS เพราะเล็งเป้าไปที่ "*ความไม่ต้องปวดหัว*" ของผู้ใช้งาน ด้วยการปล่อยให้ผู้ใช้งานทั่วๆ ไปเสี่ยงกับการต้อง "*ปวดใจ*" แทน เมื่อระบบเกิดล่มขึ้นมาจริงๆ ...

ที่สาหัสกว่านั้นก็คือ ... Microsoft Windows มีข้อจำกัดในเรื่องของการจัดแบ่ง partitions อีกต่างหาก เพราะดันไปกำหนดชื่อของ partitions ทั้งหมดด้วย "*ตัวอักษรเดี่ยว*" แล้วก็ยังกันอักษรบางตัวไว้เพื่อการใช้งานในระบบ network หรือเพื่อใช้เรียกอุปกรณ์ต่อพ่วงอื่นๆ อีกด้วย ทำให้ความยืดหยุ่นในเรื่องของการแบ่ง partitions เต็มไปด้วยข้อจำกัดที่ไม่ควรจะต้องมี จนผู้ใช้งานทั่วโลกไม่มีโอกาสได้เรียนรู้ถึงประสิทธิภาพที่แท้จริงของการจัดการระบบที่ถูกต้อง

สำหรับ MacOS ยิ่งแล้วใหญ่ เพราะตลอดประวัติศาสตร์ของ Macintosh นั้นเป็นระบบปิดที่ไม่เปิดโอกาสให้ใครเข้าไปจุ้นจ้านกับตัวระบบเลย มันคิดเองเออเองมาตั้งแต่แรก ถึงแม้ว่าจะพยายามทำให้ง่ายต่อการใช้งาน แต่ก็ไม่ได้ช่วยเพิ่มพูนความรู้ในการจัดการระบบใดๆ ให้กับ user เลย ... คำว่า partitions จึงเกือบจะไม่มีการเอ่ยถึงเลยในโลกของ Macintosh !!

ส่วนในกรณีของ Linux นั้นจะแตกต่างกัน ... *Linux ยอมให้ user แบ่ง partitions ได้อย่างอิสระ อยากจะมีเท่าไหร่ก็ทำเอาเอง อยากจะให้แต่ละส่วนใหญ่แค่ไหนก็ได้ อยากจะให้ตรงไหนเรียกว่าอะไรก็ไม่ค่อยเกี่ยงงอนมากนัก ... ปัญหาก็คือ ... จะเอายังไงกับมันดีล่ะ? เพราะถ้าแบ่งไม่ดีมันก็ติดตั้งไม่สำเร็จ หรือบางทีเราก็อาจจะเผื่อพื้นที่เอาไว้ไม่พอกับที่มันต้องการซะอีก !! ... น่าปวดขมับมาก !!*

ครั้งหนึ่งในอดีต มันจึงเกือบจะเป็นภาคบังคับที่ user ต้องอ่านเอกสารที่แนบมากับ Linux ให้เรียบร้อยซะก่อนที่จะเริ่มทำการติดตั้ง ต้องรู้จักชื่อและ spec ของอุปกรณ์ทุกชิ้นที่ตัวเองมีอยู่ ... แค่นี้ user ก็เผ่นแน่บไปแล้วครับ ยอมไปเสี่ยงกับ "*ความปวดใจ*" ที่ OS อื่นๆ หยิบยื่นให้ซะดีกว่า ... *แต่ทุกวันนี้ Linux ค่ายใหญ่ๆ อย่าง Red Hat หรือ Mandrake ก็เลิกทำตัวเป็น technicians ตาบอดกันแล้ว* มันมีระบบ auto-partitioning ให้ user ใช้งานเหมือนกับ OS ตัวอื่นๆ ด้วย ... เรียกว่าหยิบยื่นความเสี่ยงให้เลี่ยง "*ความน่าปวดหัว*" ไปทอดหนึ่งก่อน ... เพื่อความสบายใจชั่วคราวของ user นั้นๆ

โดยปกติแล้ว Linux จะใช้เวลาในการติดตั้งตัวเองประมาณ 1 ชั่วโมงครับ ซึ่งเป็น 1 ชั่วโมงที่เราจะได้ครบทั้ง OS และ application พื้นฐานที่ต้องการ ไม่เหมือนกันกับ Microsoft Windows ที่ใช้เวลาประมาณ 1 ชั่วโมงเพื่อที่จะได้แค่ OS มาปรับแต่งอะไรๆ อีกหลายอย่างก่อนที่จะเริ่มใส่ application ให้กับมัน ... ดังนั้น การเรียนรู้ Linux ก็น่าจะเริ่มจากเวลา 1 ชั่วโมงที่เราควรจะต้องจ่ายนั่นเอง ...

เราก็เพียงแต่บอกให้ Linux จัดแบ่ง partitions ตามที่มันเห็นสะดวก ... (ตรงนี้ยังไม่พูดถึงการติดตั้งเครื่องคอมพิวเตอร์แบบหลาย OS นะครับ เพราะการติดตั้งแบบนั้นควรจะต้องเรียนรู้อะไรๆ ให้มากกว่านี้อีกนิดหน่อย :-) ... เนื่องจากการทำ "*ตามสะดวก*" ของ Linux อาจจะหมายถึงการลบข้อมูลของทั้ง hard disk ทิ้งไปเลยก็ได้) ... จากนั้นก็เลือก applications พื้นฐานที่เรานึกอยากจะมีไว้ใช้งาน ... ตอบคำถามให้กับมันซักนิดหน่อย แล้วก็ปล่อยให้มันจัดการส่วนที่เหลือ สำหรับเราในฐานะของ user ก็แค่รอและคอยเปลี่ยนแผ่น CD ตามที่มันเรียกร้อง ... ประมาณ 1 ชั่วโมงให้หลัง Linux ตัวที่เราเลือกก็จะติดตั้งตัวเองจนเสร็จเรียบร้อย ... ง่ายมาก !!

ถ้าเราเป็น user ประเภท "*แดกด่วน*" เราก็จะได้ OS พร้อมด้วย applications ไว้ใช้งานบนเครื่องที่จัดหามาเรียบร้อยแล้ว ซึ่งสามารถใช้งานพื้นฐานของสำนักงานทั่วไปได้ทันทีไม่ว่าจะเป็นการพิมพ์งานเอกสารที่เรียกว่า word processor หรือสร้างกระดาษทำการที่ต้องใส่สูตรคำนวณประเภท spreadsheet, ทำ slides เพื่องาน presentation, ใช้โปรแกรมวาดรูประบายสี, ดูหนัง-ฟังเพลง, ท่อง internet หรือใช้โปรแกรม IM (Instant Messenger) ของค่ายต่างๆ รวมไปถึง e-mail ... กว่า 80% ของเครื่องคอมพิวเตอร์ประจำสำนักงานก็ใช้งานกันอยู่แค่นี้เอง ... ถูกมั้ย?

ในกรณีของ พวกมือคัน ... เรากำลังจะเริ่มบทที่ 1 กันจริง ๆ ซักที ... :-)

หลังจากการติดตั้งแบบ "*แดกด่วน*" ... Linux ได้สร้างไฟล์ขึ้นมาประมาณ 3-4 หมื่นไฟล์ แยกเก็บไว้ใน directories ต่างๆ นับ 10 directories ... ถ้าเรียกกันด้วยภาษาแบบ Linux เขามักจะเรียกทั้งหมดนั้นว่า File Systems ... **มันคือเวทมนต์ศักดิ์สิทธิ์ในโลกของ Linux ครับ ... เพราะ Linux จะจัดการทุกอย่างที่เกี่ยวกับการทำงานของมันด้วยระบบ File Systems เท่านั้น ... คือมองทุก ๆ องค์ประกอบเป็น "*ไฟล์*" อย่างสมบูรณ์ และบริหารจัดการด้วย concept ของการบริหารจัดการ "*ไฟล์*" เท่านั้นเอง !!** ... เป็นแนวคิดง่ายๆ ที่ทรงประสิทธิภาพอย่างเหลือเชื่อ ซึ่งต้องยกย่องนักคอมพิวเตอร์รุ่นบุกเบิก UNIX ที่คิดเรื่องอย่างนี้ออกมาเมื่อกว่า 40 ปีที่แล้ว และตกทอดเป็นมรดกมาถึง Linux ในปัจจุบัน

ดังนั้น แม้ว่าเราจะแบ่ง partitions ออกไปหลายๆ ส่วนตามที่เราต้องการ Linux ก็ยังคงจัดการทั้งหมดด้วยแนวคิดและวิธีการแบบ "*ไฟล์*" ธรรมดาอยู่ดี ... จะแตกต่างกันก็เพียงในด้านของ Logical ที่จะมีการกำหนดขอบเขตหรือขนาดใหญ่สุดของแต่ละ partitions ไว้ใน File System Tables (พวกเขาเรียกกันย่อๆ ว่า fstab) ...

ถ้าเรารู้ขนาดที่ต้องการใช้ในแต่ละ directories เราก็จะรู้ขนาดของ partitions ที่เราควรจะต้องจัดแบ่งไว้ตั้งแต่แรกด้วย ... ถูกมั้ย?

แต่การตัดสินใจว่าควรจะกำหนดให้แต่ละ partitions มีพื้นที่สำรองมากน้อยเท่าไหร่นั้น เราก็ต้องรู้ด้วยว่าแต่ละส่วนที่เห็นเป็น directories นั้น ถูกกำหนดให้ทำหน้าที่อะไรบ้างในระบบของ Linux

ย้อนกลับไปดูตารางหรือ "*แผนที่*" ที่ผมทำไว้ให้อีกครั้งครับ ...

File System หลักๆ ที่ Linux จะต้องมีก็คือ Root File Systems ซึ่งส่วนนี้แทนด้วยเครื่องหมาย / หรือที่เรียกว่า slash (เป็นคนละข้างกับที่เราเห็นในระบบของ DOS หรือ Windows ที่ใช้เครื่องหมาย \ หรือเรียกว่า back slash แทน) ... ถ้าเราติดตั้งแบบที่ปล่อยให้ Linux ตัดสินใจเองมันก็จะสร้างเฉพาะ partition สำหรับ / หรือ Root File Systems นี้ขึ้นมาโดดๆ แบบเดียวกับที่ Windows หรือ MacOS นิยมทำกัน

ในระบบของ Root File Systems นั้นจะต้องประกอบไปด้วย 5 directories ด้วยกัน โดยทั้ง 5 ตัวนี้จะต้องไม่แยกออกไปเป็นคนละ partitions เด็ดขาด ... ถือว่าเป็นแกนหลักของระบบ Linux ทั้งระบบเลยก็ว่าได้ครับ ... 5 directories ที่ว่านี้ก็คือ

/bin : เป็นพื้นที่ที่เก็บรวบรวมคำสั่งของ Linux สำหรับการใช้งานในระดับ user ทั่วไป ... คำว่า bin ย่อมาจากคำเต็มว่า binary โดยไฟล์คำสั่งที่อยู่ใน /bin นี้มักจะเป็น software เล็กๆ ที่ใช้งานแบบ interactive command line ล้วนๆ ... เป็นโลกที่เขาเรียกกันว่า shell หรือ terminal บางคนก็เรียกว่า text mode แบบที่จะไม่ค่อยจะได้มีโอกาสเห็นอีกแล้วในโลกของ Microsoft Windows และ MacOS

อย่างไรก็ตาม text mode ถือเป็น mode การใช้งานที่ "*บริสุทธิ์*" มากๆ ทั้งยังมีประสิทธิภาพและความคล่องตัวสูง ใช้ทรัพยากรของระบบน้อย และทำงานตรงกับเรื่องที่ต้องการอย่างรวดเร็ว ... ที่ไม่สะดวกก็คือการจดจำคำสั่งและรูปแบบการใช้งานแต่ละคำสั่ง ซึ่งมีรายละเอียดเยอะมาก ... แต่ก็คุ้มค่าสำหรับการเรียนรู้ เพราะ Linux นำคู่มือทั้งเล่ม copy ลงไปให้เราเรียกดูได้ง่ายๆ ตลอดการใช้งานอยู่แล้ว

/dev : ชื่อนี้มาจากคำเต็มว่า device มันจึงเป็นพื้นที่ที่ Linux เก็บรวบรวม drivers ของอุปกรณ์ทั้งหมดเอาไว้ตั้งแต่ hard disk, floppy disk, modem, printer, mouse, lan card, pcmcia, ฯลฯ และอื่นๆ ... เป็นจุดรวมที่ Linux จะใช้ค้นหาวิธีการสื่อสารกับอุปกรณ์ทั้งหมดที่ต่อพ่วงอยู่กับเครื่องคอมพิวเตอร์ของเรา

/etc : ผมแปลคำนี้ว่า "*จับฉ่าย*" หรือที่ภาษาไทยเราใช้เครื่องหมายเป็น "*ฯลฯ*" นั่นแหละครับ ภาษาอังกฤษอ่านว่า et cetera หมายถึง "*อื่นๆ อีกมากมาย*" ... เป็น directory ที่เก็บไฟล์ configuration ทั้งหมดของระบบเลยครับ ... เรียกว่าจะตกแต่งดัดแปลงอะไรให้กับระบบก็ให้มาจัดการเขียนคำสั่งทิ้งไว้ในนี้ก็แล้วกัน ไม่ว่าจะเป็นการติดตั้งระบบ network, web server, mail server, ระบบรักษาความปลอดภัยประเภท firewalls, และอื่นๆ ... Linux ที่ว่าใช้งานได้ตั้งหลากหลายรูปแบบนั้น เกือบทั้งหมดครับที่กำหนดวิธีทำงานให้กับมันจากใน directory นี้เพียงแห่งเดียวเลย

/lib : มาจากคำเต็มว่า library ซึ่งเป็นแหล่งรวมของ module ย่อยๆ อีกมากมายให้เราเลือกใช้งานควบคู่ไปกับระบบของ Linux ... ความยืดหยุ่นของ Linux ก็อยู่ที่ตรงนี้ด้วยครับ เพราะ Linux ทำงานในลักษณะที่เป็น module มาประกอบกัน โดย user จะสามารถเลือกได้ว่าต้องการให้ Linux เรียกเฉพาะ module ไหนขึ้นมาใช้งาน หรืองดการเรียกใช้ module ไหนที่ไม่มีความจำเป็น ... ด้วยวิธีการอย่างนี้ Linux จึงสามารถที่จะติดตั้งให้เป็นเครื่องแม่ข่าย (server) หรือเครื่องลูกข่ายแบบ workstation ได้โดยไม่ต้องแยก software ออกเป็นคนละผลิตภัณฑ์ (การแยก product ออกจากกันอย่างในกรณีของ Red Hat นั้นเป็นเรื่องของการให้บริการเท่านั้นครับ Linux ตัวที่เราได้มาคือตัวเดียวกันแต่ถูกกำหนดค่าให้เรียกใช้งานแต่ละ module ไว้แตกต่างกัน หรืออาจจะจัดหา modules ไว้ให้มากน้อยแตกต่างกันบ้างเท่านั้น ซึ่ง user พอจะสามารถหาจากแหล่งอื่นๆ ได้ฟรีอยู่แล้ว

/sbin : เป็นไฟล์ binary สำหรับ superuser หรือผู้ดูแลระบบเท่านั้นครับ ... ส่วนนี้น่าจะนับเป็นสุดยอดของความปลอดภัยที่คนออกแบบ UNIX และ Linux คิดออกมาเลยทีเดียว คำสั่งใช้งานที่อันตรายแก่ระบบทั้งหมดจะถูกแยกเก็บต่างหากจาก /bin และไม่อนุญาตให้ user ทั่วไปมองเห็นไฟล์คำสั่งเหล่านี้เลยด้วย ... เพราะฉะนั้นการที่ user ธรรมดาๆ คนหนึ่งจะทำอะไรให้ระบบถึงขั้นเสียหายหรือล่มลงไปจริงๆ นั้นไม่ใช่เรื่องง่ายเลยในโลกของ Linux

ปกติแล้วทั้ง 5 directories นี้ user จะได้รับสิทธิ์ใน "*การอ่าน*" หรือเรียกไปใช้งานเท่านั้น แต่ไม่มีสิทธิ์แก้ไขหรือดัดแปลงใดๆ ทั้งสิ้น เพราะต้องปล่อยให้เป็นธุระของคนที่มีหน้าที่ดูแลระบบโดยตรงเพียงคนเดียว ... โดยทั่วไปแล้ว Root File Systems ที่ถือเป็นหัวใจของระบบนั้น เกือบจะไม่มีการเพิ่มเติมหรือเปลี่ยนแปลงในด้านขนาดของการจัดเก็บอีกเลยหลังจากการติดตั้ง ... ดังนั้น ส่วนใหญ่เขาจึงจัดการแบ่ง partition ออกมาเพื่อทำหน้าที่ของ Root File Systems แค่เพียงพอกับการ upgrade ไว้บ้าง หรือเผื่อการหา module มาเพิ่มภายหลังเล็กๆ น้อยๆ เท่านั้น โดยจะไม่ยอมให้ปะปนกับพื้นที่ส่วนอื่นๆ เพื่อให้การเปลี่ยนแปลงแก้ไขหรือการติดตั้งเพิ่มเติมแก่ระบบ สามารถแยกเป็นเอกเทศต่างหากจากไฟล์ข้อมูลหรือ application พื้นฐานที่มีการติดตั้งไว้ทั้งหมด

boot images

มี directory เล็กๆ อยู่ตัวหนึ่งที่น่าจะพูดถึงครับ เราจะเห็นมันแสดงตัวในชื่อว่า /boot ซึ่งโดยปกติมันจะมีขนาดรวมเบ็ดเสร็จแล้วประมาณ 5 MB เท่านั้นเอง

ข้างในของ /boot จะเก็บไฟล์ที่เรียกกันว่า boot images หรือ kernel images เอาไว้ ซึ่งเป็นไฟล์ที่ต้องใช้ในการ boot ระบบ Linux ขึ้นมา เรียกว่าถ้าส่วนนี้เสียหายก็เป็นอันว่าไม่ต้อง boot ขึ้นมาใช้งานกันล่ะ เพราะนี่คือส่วนที่เป็น kernel ที่จะกำหนดค่าต่างๆ ให้เครื่องคอมพิวเตอร์ของเราสามารถเข้าใจคำสั่งแบบ Linux ที่เก็บอยู่ใน Root File Systems ได้นั่นเอง

จริงๆ แล้วมันต้องทำงานคู่กับ Root File Systems อยู่แล้วครับ บางครั้งจึงไม่นิยมแยกออกไปเป็น partition ต่างหากจาก Root File Systems แต่ผู้ดูแลระบบหลายคนก็ยังนิยมแยกมันออกไป รวมทั้งการติดตั้ง Linux ก็ยังมี option ให้เราสามารถแยก partition นี้ออกมาจาก Root File Systems ได้ด้วย ... เพราะโดยปกติแล้ว Linux มักจะมีการปรับปรุงแก้ไขเฉพาะส่วนของ kernel เล็กๆ น้อยๆ ก่อนที่จะมีการ upgrade ครั้งใหญ่ๆ ในระดับ version หรือตัวเราเองสามารถที่จะเข้าไปกำหนดค่าของ module ต่างๆ แล้วสร้าง kernel ที่ใช้งานอย่างเฉพาะเจาะจงของเราขึ้นมาเองก็ได้ ... การแยกออกมาต่างหากก็เพื่อให้มันสะดวกกับการซ่อมแซมเฉพาะส่วนเฉพาะเรื่องของการ boot โดยที่ไม่ต้องเริ่มติดตั้งระบบใหม่เท่านั้นเอง

เรื่องของ boot images มีส่วนที่ใกล้ชิดกับเรื่องของ boot sequence หรือลำดับขั้นตอนในการ boot ระบบ ซึ่งเป็นส่วนที่ 2 ของเอกสารชุดนี้ เพราะฉะนั้นจึงขอเว้นไว้เล่ารายละเอียดทีหลังครับ

/usr : พื้นที่สำหรับเก็บ software และ application ต่าง ๆ

ถ้าเปรียบเทียบกับ Microsoft Windows แล้วนี่ก็เปรียบเสมือน folder ที่ชื่อว่า Program Files นั่นเอง ... เป็นแหล่งรวมของ software และ application ต่างๆ เพื่อการใช้งานที่นอกเหนือจากการดูแลระบบด้วย shell command

การติดตั้ง Linux มักจะนิยมให้แยก /usr ออกไปเป็น partition ต่างหาก เพราะเป็นส่วนที่ user หรือ admin สรรหาอย่างอื่นเข้ามาเพิ่มเติมให้กับการใช้งาน แต่ไม่ถึงกับเป็นตัวระบบจริงๆ ... ถึงแม้ว่าจะมี application ที่มีส่วนเกี่ยวข้องกับระบบหลายตัวถูกจัดเก็บไว้ที่ /usr นี้ก็ตาม แต่โดยปกติของการใช้พื้นที่ส่วนนี้ ก็มักจะโตวันโตคืนพอสมควร ขึ้นอยู่กับความขยันหา application แปลกๆ มาเพิ่มเข้าไปมากแค่ไหนด้วย ... ดังนั้น เพื่อเป็นการป้องกันไม่ให้เกิดการเติมจนเพลิน แล้วลุกลามไปกินพื้นที่ส่วนอื่นๆ ของระบบเข้า เขาจึงนิยมให้แยกไว้ต่างหาก

/var : พื้นที่เพื่อการสื่อสารและการตรวจสอบ

ไม่แน่ใจเหมือนกันว่ามันจะมาจากตัวเต็มๆ ว่า varieties รึเปล่า? เพราะส่วนใหญ่ของไฟล์ที่ถูกจัดเก็บอยู่ใน /var นี้มีความหลากหลายไม่แพ้ /etc เหมือนกัน คือเป็นทั้ง mail box, message log, log file, ตลอดไปจนถึงเรื่องของ web server ด้วย ... ผมจึงเหมาๆ เอาว่ามันเป็นพื้นที่ที่ Linux ใช้เพื่อ "*การสื่อสารในระบบ*" มากกว่าจะเป็นการเก็บไฟล์ข้อมูลที่ต่อเนื่องเป็นชิ้นเป็นอัน

หัวใจสำคัญของ /var ที่เกี่ยวข้องกับการดูแลรักษาระบบก็คือ มันจะมีไฟล์ที่บันทึกร่องรอยของการทำงานทั้งหมดของ Linux เอาไว้ตลอดเวลา เพื่อให้เราสามารถเข้ามาค้นดูได้ว่า ปัญหาที่เกิดขึ้น ณ เวลาหนึ่งๆ นั้น มีสาเหตุมาจากส่วนไหนของระบบบ้าง เพื่อที่เราจะได้ตามไปแก้ไขให้ถูกทิศถูกทาง ... ตรงนี้ก็ต้องถือเป็นส่วนหนึ่งของการสื่อสารที่ระบบติดต่อกับผู้ดูแลอยู่ตลอดเวลา

นอกจากนั้นแล้ว /var ก็มีส่วนที่เอาไว้เก็บ spool ต่างๆ เช่นการสั่งพิมพ์ผ่าน print server และระบบ e-mail ที่จะพักไฟล์ข้อความระหว่าง user ทุกคนไว้ใน mailbox ที่อยู่ใน /var นี้เหมือนกัน

สำหรับ web server ซึ่งเป็นเรื่องของการกระจายข่าวสารข้อมูลแบบไม่เจาะจงผู้รับ ผมก็ยังถือว่าเป็น

เรื่องของการสื่อสารอีกเช่นกัน แม้ว่าเราสามารถกำหนดให้ web pages ของเราอยู่ที่ไหนก็ได้ในระบบ แต่ค่า defaults ของ Linux ก็คือเก็บอยู่ภายใต้ directories ใน /var นี่เท่านั้น

อย่างไรก็ตาม โดยหน้าที่จริงๆ แล้ว /var เกือบจะถือว่าไม่มีความเกี่ยวข้องกับการใช้งานโดยตรง แต่มีประโยชน์เพื่อการดูแลรักษาและการตรวจสอบการใช้งานของระบบเท่านั้น องค์กรใหญ่ๆ หลายแห่งจึงมีการสั่ง backup ข้อมูลประเภทนี้แยกออกไปเก็บต่างหากเป็นระยะๆ ... การแยก partition ต่างหากให้กับ /var จึงเป็นเรื่องของความสะดวกในการ backup มากกว่าการป้องกันความเสียหายที่อาจจะเกิดขึ้น

/tmp : กระดาษทดของระบบ

เป็น directory ยอดนิยมครับตัวนี้ ย่อมาจาก temporary ที่หมายถึง "*ชั่วคราว*" ส่วนใหญ่จะเป็นไฟล์ขยะที่เกิดขึ้นระหว่างกระบวนการทำงานของ application ต่างๆ หรืออาจจะเกิดจากคำสั่งบางอย่างของระบบชะเอง

ไฟล์ประเภทนี้ไม่มีความจำเป็นต้อง backup แล้วก็ไม่จำเป็นต้องกลัวเสียหรือหาย แต่ที่ต้องกลัวคือมันจะเลอะเทอะอยู่ใน File Systems อื่นๆ เท่านั้นเอง ... ดังนั้น /tmp จึงควรจะถูกแยกออกไปเป็น partition ต่างหาก เราจะได้ซัดกับมันได้เต็มที่ไม่ว่าในกรณีใดๆ ก็ตามครับ

swap : หน่วยความจำสำรอง

ตัวนี้เป็น File System ต่างหากประเภทหนึ่งไปเลยครับ และไม่ได้ถูกมองเป็น directory เพราะปกติจะมีโครงสร้างของการจัดเก็บไม่เหมือนระบบไฟล์ปกติของ Linux ... ดังนั้นจึงควรจะแยกออกไปเป็น partition ต่างหาก เพื่อให้มันสามารถถูก format ให้เป็นโครงสร้างแบบพิเศษของมัน (มีบางกรณีที่เราไม่สามารถสร้าง swap partition ขึ้นมาในระบบที่เราติดตั้ง เราก็สามารถสร้างไฟล์เปล่าขนาดใหญ่ๆ ขึ้นมาทำหน้าที่นี้แทนก็ได้ ซึ่งจะคล้ายกับ Microsoft Windows ที่ไม่ได้แยก swap file ออกไปเป็น partition ต่างหาก)

หน้าที่หลักของ swap ก็คือ มันจะทำตัวเป็น "*หน่วยความจำสำรอง*" ของระบบ ในกรณีที่หน่วยความจำหลักที่เป็น memory card แบบ hardware จริงๆ เกิดไม่พอใช้ขึ้นมา Linux ก็จะทดลงไปในส่วน swap นี้ โดยสิ่งที่แตกต่างจาก /tmp ก็คือ อะไรก็ตามที่ถูกเขียนลงไปใน swap จะถูกลบหายไปเองเมื่อความจำเป็นที่จะต้องใช้งานนั้นหมดไป โดยที่เราไม่ต้องตามไปเช็ดหรือตามไปลบให้วุ่นวาย

/home : พื้นที่ส่วนตัวของ user

ถ้าจะพูดถึงไฟล์ขยะ พื้นที่ใน /home นี้เองที่น่าจะมีขยะมากที่สุด และเป็นขยะที่ไม่ค่อยจะมีใครกล้าทำลายแบบสุ่มสี่สุ่มห้าด้วย เพราะส่วนใหญ่จะเป็นไฟล์ข้อมูลที่ user แต่ละคนจัดเก็บเอาไว้ แต่อาจจะรกร้างและไม่ได้รับการดูแลรักษาเท่าที่ควร ... ดังนั้น /home จึงเป็นอีกหนึ่ง partition ที่สมควรจะแยกออกไปต่างหากมากที่สุด เพราะนอกจากมันจะไม่ถูกรบกวนเนื่องจากการติดตั้งระบบใหม่แล้ว มันยังอาจจะมีการกำหนด quota ของพื้นที่ใช้สอยสำหรับ user แต่ละชื่อได้อีกด้วย (*การกำหนด **quota** ของ **Linux** เป็นการกำหนดในระดับ **partition** ดังนั้นพื้นที่ส่วนไหนที่จำเป็นต้องมีการกำหนด **quota** ของ **user** จึงต้องแยกออกไปเป็น **partition** ต่างหากเสมอ*)

พื้นที่ใน /home ถือเป็นพื้นที่ของ user โดยตรงที่จะได้รับสิทธิ์เต็มที่ในการจัดการไฟล์ข้อมูลที่อยู่ภายใต้ชื่อของตัวเองทั้งหมด ซึ่ง user จะต้องหมั่นดูแลรักษาอย่าให้รกเลอะเทอะหรือมีขยะเก็บไว้มากจนเกินไป ***เพราะในการเรียกใช้งาน application บางตัว หรือระบบ graphic ของ Linux ที่***

เรียกกันว่า X Windows นั้น user หนึ่ง ๆ จะต้องสามารถเขียนไฟล์บางอย่างลงไปใน /home ของตัวเองได้ด้วย มิฉะนั้นแล้ว ก็จะไม่สามารถเข้าสู่ graphic mode ของ Linux ได้เลย

ทั้งหมดนี้ก็คือ partitions หลักๆ ของ Linux ที่พวกเราจำเป็นจะต้องรู้จักเอาไว้ และจะต้องกำหนดขนาดของมันให้พอเพียงกับการใช้งานจริงด้วย เพื่อให้ระบบของเรามีประสิทธิภาพและมีความยืดหยุ่นบนพื้นฐานที่จะต้องปลอดภัยที่สุด ... ปัญหาของคนที่เพิ่งจะเคยสัมผัสกับ Linux ก็คือ มันไม่มีกฎตายตัวในเรื่องของขนาดสำหรับแต่ละ partitions เพราะขึ้นอยู่กับความจำเป็นของเราเอง ... ผมก็เคยใช้วิธีการลงแบบ auto-partitioning เพื่อขอดูขนาดแต่ละ directories แล้วนำมาประกอบการตัดสินใจก่อนที่จะติดตั้งระบบเป็นรอบที่สองหรือสามสี่ห้า ... ซึ่งในขณะที่เราไม่ได้คุ้นเคยหรือชำนิชำนาญกับการดูแลระบบด้วยตัวเอง ผมคิดว่าเราน่าจะลองศึกษาตัวเลขที่ตำราบางเล่มแนะนำเอาไว้เป็นแนวทางเบื้องต้นก็น่าจะเป็นความคิดที่ไม่เลวนัก ... ปกติเขาจะแนะนำกันอย่างนี้ครับ ...

/	: 256 MB
/boot	: 5 MB
<swap>	: 2 เท่า -2.5 เท่า ของขนาด physical memory
/usr	: 512 MB
/var	: 256 MB
/tmp	: 329 MB
/home	: ประมาณ 100 MB ต่อ 1 user

ดังนั้น ถ้าสมมุติว่าเรามี physical memory อยู่ที่ 256 MB เราก็ควรจะมี swap partition ใหญ่ประมาณ 512 MB และหากเราเตรียมเครื่องไว้ใช้งานร่วมกันซัก 5 user ที่ /home ก็ควรจะมีขนาดประมาณ 500 MB ... *หลงจ้ง*ก็จะต้องใช้พื้นที่ hard disk ประมาณ 2.37 GB เพื่อติดตั้ง Linux ทั้งตัวพร้อม application พื้นฐานครับ

ตรงเรื่องของ swap partition นั้นมีเกร็ดเล็กๆ ทางเทคนิคเล่าว่า ขนาดใดๆ ที่ใหญ่เกินกว่า 256 MB นั้นจะเป็นพื้นที่เสียเปล่าที่ไม่มีโอกาสถูกใช้งานเลยในโลกของ Linux !? ... แต่ทุกตำราก็จะพยายามบอกไว้เหมือนๆ กันคือให้สร้าง swap partition ประมาณ 2-2.5 เท่าของ physical memory ... ทั้งๆ ที่ เครื่องคอมพิวเตอร์ที่มีหน่วยความจำระดับ 256 MB นั้น Linux เกือบจะไม่เคยถามหา swap เลยซักครั้งเดียว !! ... ถ้างั้นอะไรคือเหตุผลของการสร้าง swap partition ไว้ใหญ่โตเกินจำเป็น?

เกี่ยวกับเรื่องนี้ ผู้รู้ทางด้าน Linux กล่าวว่า swap ส่วนที่เกินจำเป็นนั้น จะถูกใช้งานในกรณีเดียวคือการ dump ข้อมูลของทั้ง hard disk เพื่อเหตุผลในการทำสำเนาก่อนที่จะมีการตรวจซ่อมบำรุงเครื่องแม่ข่ายหรือ Server เพราะโดยปกติแล้ว Linux จะไม่ได้เขียนไฟล์จำนวนมากๆ พร้อมๆ กันในลักษณะนั้น เราจึงไม่เคยพบเห็นความต้องการ swap ที่ใหญ่เกินกว่า 256 MB เลยตลอดการใช้งาน

อย่างไรก็ตาม ในกรณีของคนที่กำลังคิดที่จะเริ่มแกะรอย Linux ส่วนใหญ่ก็มักจะนำเครื่องรุ่นเก่าๆ มาติดตั้งระบบเพื่อศึกษาเรียนรู้ ... spec เครื่องจึงอาจจะต่ำขนาดว่ามี hard disk อยู่เพียงไม่เกิน 2 GB ซ้ำยังมี physical memory ไม่กี่สิบ MB อีกต่างหาก ... ก็ต้องขอแสดงความยินดีด้วยครับ เพราะ Linux สามารถถูกติดตั้งในลักษณะที่บีบๆ รัดๆ อย่างนั้น ด้วย options ของระบบที่อาจจะไม่ full features ได้ด้วยเหมือนกัน ... พื้นที่ของ partitions ต่างๆ ก็จะมีความจำเป็นด้านขนาดเล็กลงตามตัวเลขนี้ครับ ...

/	: 35 MB
/boot	: 5 MB
/usr	: 232 MB
/var	: 25 MB
/tmp	: 30 MB
/home	: 100 MB

นี่คือ minimum ที่ Linux ต้องการครับ ... รวมกันทั้งหมดยังไม่ถึง 500 MB เลยด้วยซ้ำ!! ซึ่งก็ต้องยอมรับความจริงด้วยว่าการติดตั้งในพื้นที่จำกัดจำเขี่ยอย่างนี้ ไม่ต้องหวังผลในเรื่องของ graphic ครับ เพราะมันเป็นไปไม่ได้เลย ... Linux ที่ติดตั้งในสภาพแวดล้อมอย่างนี้จึงทำงานได้เฉพาะ text mode หรือ command line เท่านั้นเอง ... แต่ก็เหมาะกับการศึกษาเรียนรู้ และใช้เครื่องคอมพิวเตอร์รุ่นเก่าๆ ให้เป็นประโยชน์มากที่สุดด้วยครับ เพราะงาน system admin ส่วนใหญ่ไม่ได้มีความต้องการในด้าน graphic mode เลย (นอกจากนั้นก็ยังมีข้อแนะนำว่า ไม่ควรที่จะใช้ชื่อของ system admin หรือ root ในส่วนทีเป็น graphic mode อีกต่างหาก !!)

ขออนุญาตเล่าเปรียบเทียบกับ Microsoft Windows ซักเล็กน้อยครับ การแบ่ง partitions แบบที่ยกมาให้เห็นนี้ ถือว่าเป็น minimum ที่ควรจะเป็นในโลกของ Linux ... แต่ก็เป็นสิ่งที่แทบไม่น่าจะเกิดขึ้นได้ในโลกของ MS Windows เพราะ MS Windows จะต้องกำหนด "*ตัวอักษร*" 1 ตัวให้กับแต่ละ partitions ที่มีในระบบของมัน ... ดังนั้นเราจึงน่าจะเห็น MS Windows ทำอย่างนี้ครับ

/ : ให้เป็น drive C:
/boot : ให้เป็น drive D:
/usr : ให้เป็น drive E:
/var : ให้เป็น drive F:
/tmp : ให้เป็น drive G:
/home : ให้เป็น drive H:

ซึ่งถ้ามีการติดตั้ง CD-drive ไว้ มันก็ต้อง run ชื่อต่อไปอีก เป็น I:, J:, K:, ... etc. ... มี external drives หรืออุปกรณ์บันทึกข้อมูลอะไรที่ต่อพ่วงอยู่ ทั้งหมดก็จะถูกกำหนดชื่อให้ต่อเนื่องไปเรื่อยๆ อย่างนี้ ... แต่ก็เป็นไปได้ "*ในทางทฤษฎี*" เท่านั้นครับ เพราะ MS Windows จำเป็นต้องสงวนบาง drive ไว้ใช้เฉพาะงานของมันอย่างเช่น A: กับ B: ที่หัวเด็ดตีนขาด MS Windows ก็ไม่ยอมรับให้มันเป็นส่วนหนึ่งของ hard disk อย่างนี้เป็นต้น ... นี่ก็เป็นอีกเหตุผลหนึ่งที่นักคอมพิวเตอร์หลายๆ คนมักจะรู้สึกว่า MS Windows ไม่ใช่ OS ในแบบที่พวกเขาต้องการอีกต่อไป

ทั้งหมดที่เล่ามาเป็น "*แนวความคิด*" หรือ concept หลักๆ เกี่ยวกับเรื่อง partitions เท่านั้นครับ ยังไม่ได้พูดถึงคำสั่งอะไรเลยซักคำสั่งเดียว ... เพราะการศึกษาเรียนรู้คอมพิวเตอร์ที่ถูกต้องนั้น เราจะต้องเข้าใจใน concept ของมันให้ถูกต้องซะก่อน เพื่อเป็น "*ฐานราก*" ในการรองรับระดับความรู้ และรายละเอียดอื่นๆ ที่จะต้องซ้อนทับจนสูงขึ้นเรื่อยๆ

การแบ่ง Partitions ของ Linux

คำสั่งหรือ software ที่เข้ามาเกี่ยวข้องกับการจัดการ partitions ของ Linux มีอยู่ด้วยกัน 3 ตัวครับ คือ Disk Druid, fdisk และ parted ... ส่วนคำสั่ง partprobe จะเป็นแค่คำสั่งเสริมเฉยๆ

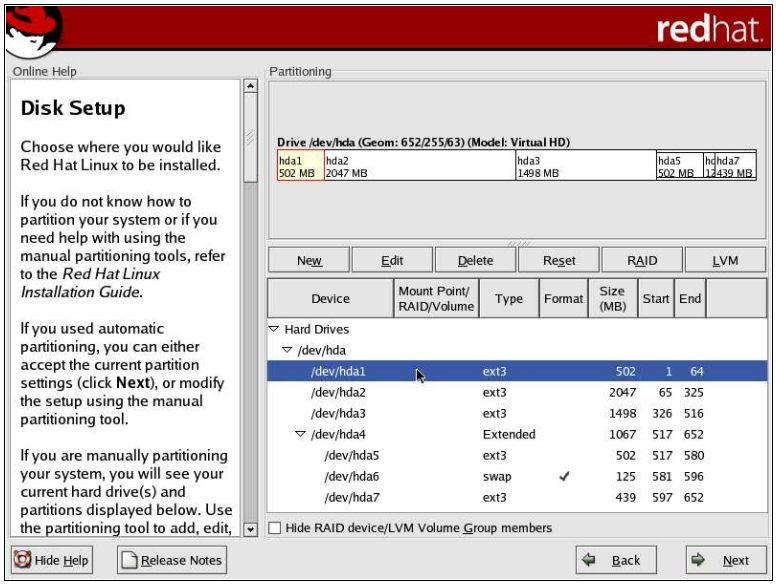
Disk Druid เป็นชุดคำสั่งที่ทำงานได้ทั้งใน graphic mode และยังสามารถทำงานแบบ menu driven ใน text mode ได้ด้วย แต่ก็เฉพาะตอนที่จะทำการติดตั้งครั้งแรกเท่านั้น หมายความว่าพวกเราทุกคนจะมีโอกาสได้พบเห็นกับ Disk Druid เพียงครั้งเดียว แล้วก็จะไม่ต้องเจอหน้าเจอตากันอีกตลอดไปจนกว่าจะมีการติดตั้งระบบใหม่ ... รูปแบบการใช้งาน Disk Druid นั้นถือว่าง่ายมากทีเดียว มีการใช้ mouse เป็นอุปกรณ์ในการสั่งงานเกือบทั้งหมด ชื่อ Partitions หลักๆ ก็มีให้เลือกเป็น options อยู่แล้วโดยที่ user ไม่จำเป็นต้องจำตัวสะกดให้วุ่นวาย

โดยส่วนมากแล้ว คู่มือ Linux มักจะบอกเล่าเฉพาะเรื่องของ fdisk เพียงคำสั่งเดียว เพราะเป็นคำสั่งที่

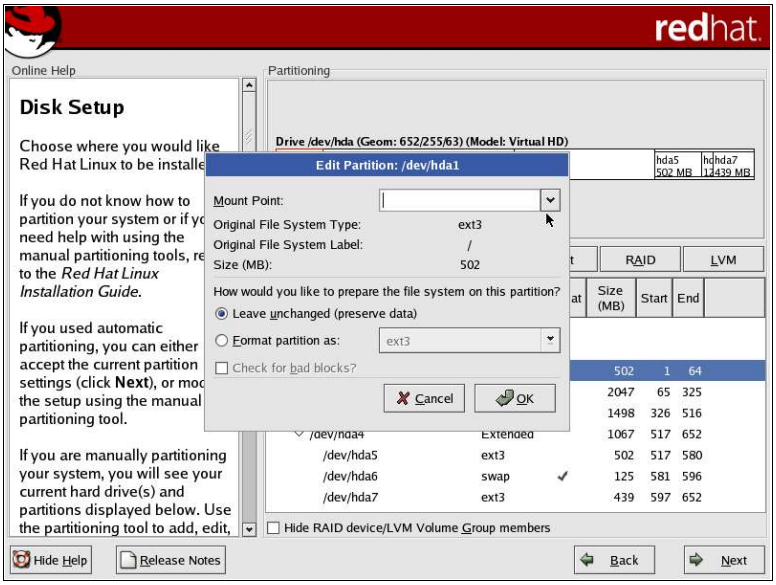
เราใช้เพื่อการจัดการ partition โดยที่ไม่จำเป็นต้องผ่านกระบวนการติดตั้งใหม่ รวมทั้ง user ที่เรียนรู้รายละเอียดต่างๆ จาก fdisk ไปแล้ว ก็น่าที่จะเข้าใจ Disk Druid ได้ในทันทีด้วย เพราะส่วนที่แตกต่างกันจริงๆ ก็จะมีเพียง interface ของโปรแกรมที่ใช้งานเท่านั้น

อีกเหตุผลหนึ่งที่ตำราทั้งหลายไม่นิยมเอาเรื่องของ Disk Druid มาเขียนก็เพราะว่า Linux นั้นมีอยู่ด้วยกันหลายเผ่าหลายตระกูลพอสมควร และส่วนที่เป็น graphics ก็จะมีการออกแบบที่แตกต่างกันไป ทำให้คู่มือต่างๆ ไม่สามารถเจาะจงวิธีการหรือขั้นตอนต่างๆ ได้อย่างสะดวก ... แตกต่างกับกรณีของคำสั่ง fdisk ซึ่งเป็น text mode ที่เหมือนๆ กันหมดทุกค่ายของ Linux

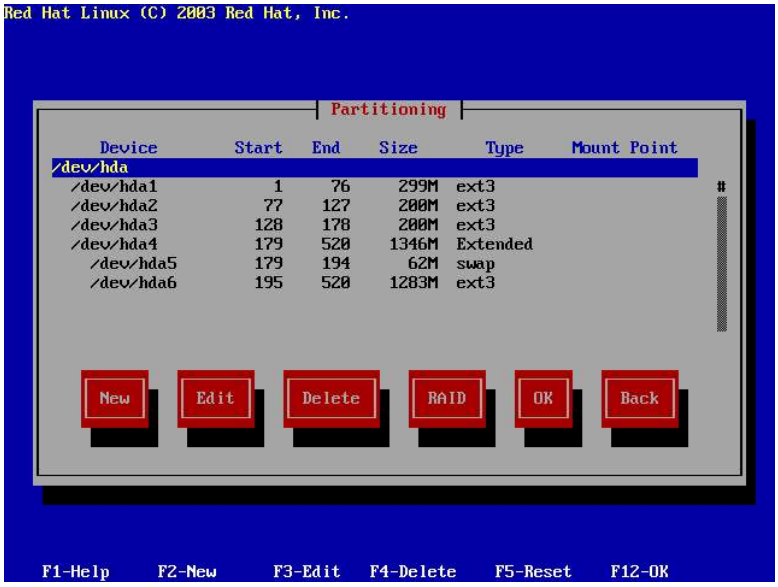
หน้าตาของ Disk Druid ที่มาจากค่าย Red Hat จะมีให้เห็นได้ทั้ง 2 mode อย่างนี้ครับ



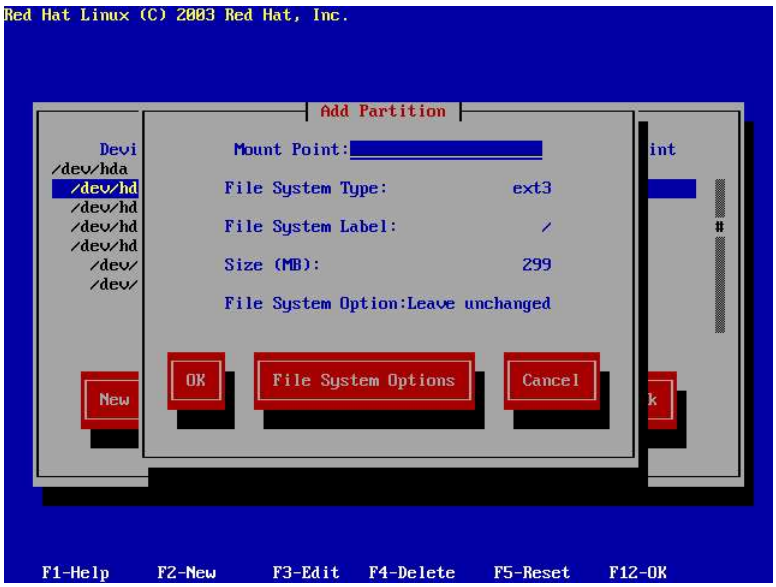
นี่คือการ dump หน้าจอ Disk Druid จากเครื่องคอมพิวเตอร์ที่เคยมีการแบ่ง partitions ไว้ก่อนแล้วใน hard disk จะเห็นว่ามันมี menu คำสั่งอยู่ตอนกลางๆ จอให้เราเลือกใช้ ด้านบนเป็น graphic ง่ายๆ เพื่อบอกเราว่าพื้นที่ hard disk นั้นๆ ถูกใช้ไปมากน้อยเท่าไหร่แล้ว ยังเหลือพื้นที่ว่างๆ อีกรึเปล่า ด้านล่างของจอก็เพียงแต่บอกเรื่องเดิมในแบบตารางรายงาน และให้รายละเอียดของ file system ที่ถูกเลือกใช้ในแต่ละ partition ด้วย ในกรณีที่เราเลือกคำสั่ง Edit เราก็จะเห็นหน้าจอถัดมานี้ครับ



ทีนี้ถ้าเป็น text mode บ้างล่ะ? ... ของค่าย Red Hat ก็จะออกมาเป็นอย่างที่เห็นข้างล่างนี้ครับ



แล้วถ้าเลือกใช้คำสั่ง Edit จาก menu ที่เห็น Disk Druid ในแบบ text mode ก็จะแสดงหน้าจอออกมาอย่างที่เห็นต่อไปนี้ครับ



ก่อนที่จะไปเล่าต่อที่คำสั่ง fdisk เพื่ออธิบายขั้นตอนในการแบ่ง partitions สำหรับการติดตั้ง Linux ขอแทรกเรื่องของ hard disk ในโลกของ Linux ไว้ย่อๆ ตรงนี้ซักหน่อยหนึ่ง ...

ย้อนกลับไปดูรูปที่ dump มาให้อีกครั้ง เราจะเห็นว่า Linux เรียกชื่อ hard disk ของเครื่องคอมพิวเตอร์ด้วยคำว่า /dev/hda ... จากนั้นมันก็จะระบุลำดับของ partitions ต่อๆ กันไปเรื่อยๆ เป็น /dev/hda1, /dev/hda2, /dev/hda3. ... จนกว่าจะครบทุก partitions ที่เราแบ่งให้กับมัน ...

ถ้ายังจำกันได้ /dev เป็นส่วนหนึ่งของ root filesystem นะ เป็นพื้นที่ส่วนที่ Linux ใช้เก็บ drivers ของอุปกรณ์ต่างๆ เอาไว้ทั้งหมด ซึ่งก็รวมทั้ง hard disk ด้วย ... **โดยในปัจจุบัน Linux จะรู้จัก hard disk อยู่ 2 ตระกูลคือ IDE และ SCSI** ... drivers สำหรับกลุ่ม IDE ก็จะเป็นพวกที่มีชื่อเรียกเป็น /dev/hd? เช่น hard disk ตัวแรกก็จะเป็น /dev/hda, ตัวที่ 2 ก็จะเป็น /dev/hdb, ตัวที่ 3 ก็เป็น /dev/hdc, ... เรียงลำดับตามอักษรไปเรื่อยๆ ... แต่ถ้าเป็น hard disk กลุ่มที่เป็น SCSI ก็ต้องใช้ drivers เป็น /dev/sd? เช่น /dev/sda, /dev/sdb, /dev/sdc, ... เรียงไปอย่างนี้เหมือนกัน

เพราะฉะนั้นเวลาที่เราเห็นชื่อ partitions ต่างๆ ในระบบของ Linux เราก็พอจะบอกได้ว่า hard disk ที่ใช้งานอยู่นั้นเป็นตระกูลไหน แล้วแต่ละ partition นั้นอยู่บน hard disk ลูกเดียวหรือว่าแยกกันอยู่หลายลูก ... อย่างในกรณีที่เห็นตัวอย่างข้างบนนั้น จะเห็น /dev/hda1, /dev/hda2, ... ไปเรื่อย ก็แปลว่ามันเป็น partitions ที่แบ่งใน hard disk แบบ IDE ลูกเดียวกันทั้งหมดเลย ... อย่างนี้เป็นต้น

ในโลกของ Linux นั้น Partitions แบ่งเป็น 2 ประเภทคือ Primary Partitions กับ Extended Partition ... ทั้งนี้จำนวนของ Primary Partitions จะมีได้สูงสุดเพียง 4 partitions เท่านั้น ถือเป็นข้อจำกัดของ hard disk *ที่ถูกควบคุมการทำงานด้วย chip ในตระกูลของ intel* กับ Linux ในปัจจุบัน *(รายละเอียดเกี่ยวกับ chip ในแต่ละตระกูลนั้นผมเองก็ไม่มีรายละเอียดมากนัก เพราะส่วนใหญ่จะใช้งานกับเครื่องในตระกูลของ intel เพียงตระกูลเดียว และถือเป็นตระกูลต้นกำเนิดของ Linux ด้วย เพราะ Linux ที่ได้รับการดูแลโดยตรงจาก Linus Torvalds คือ Linux บน platform ของ intel เพียงตระกูลเดียว ส่วน Linux บน chip แบบ PowerPC จะได้รับการดูแลโดย Cort Dougan แล้ว Alan Cox ก็จะดูแลในส่วนของ MIPS)* ... ดังนั้นหากเราต้องการแบ่ง partitions ให้มากกว่านั้น ... อย่างที่เห็นในตัวอย่าง ... เราก็ต้องยอมสละ Primary Partition ตัวหนึ่งมาเป็น Extended Partition จากนั้นค่อยแบ่งพื้นที่ใน Extended Partitions ออกเป็นส่วน

ย่อยๆ ต่อไปได้อย่างไม่จำกัด ... ผมไม่แน่ใจว่า OS อื่นๆ แก้ปัญหาเหมือนอย่างนี้มั้ย เพราะ MS Windows รุ่นแรกๆ จะยอมให้มีแค่ 1 Primary กับ 1 Extended เท่านั้น (แล้วไปแยกเป็น drives ต่างๆ นอกเหนือจาก C: ใน Extended Partition) และจำกัดว่า OS จะต้องอยู่ที่ Primary Partition เท่านั้นอีกด้วย ... แต่ Linux ไม่ได้จำกัดว่า OS ต้องอยู่ที่ไหน โดยมันจะถือว่าทุก partitions ที่แบ่งออกมามีศักดิ์ศรีเทียบเท่ากันเสมอ ... บทบาทที่แตกต่างกันของแต่ละ partitions จะขึ้นอยู่กับ user ที่ติดตั้งระบบว่าต้องการให้ partitions ไหนทำอะไรบ้าง ... ซึ่งในโลกของ Linux จะเรียกการกำหนดบทบาทของ แต่ละ partitions ในลักษณะนี้ว่า "*การกำหนด mount point*"

แต่ละ mount point ของ Linux อาจจะเรียกได้อีกอย่างว่า file system โดยมี mount point ภาคบังคับที่จะต้องมีก็คือ Root File Systems ที่แทนด้วยสัญญลักษณ์ / นั่นเอง ... สำหรับ mount point อื่นๆ ที่เล่าไปแล้วคือ /boot, /usr, /var, /tmp, /home ก็จะถูกสร้างเป็นเพียง directories ที่ย่อยลงไปจาก / หรือ Root File Systems เท่านั้นหากเราไม่แยก partitions ออกไปต่างหากแล้วกำหนดเป็น mount point ให้กับพวกมัน

เอาล่ะ ... ผมถือโอกาสดึงเอาหน้าจอของ Disk Druid ออกมาเพื่อเป็น "*ตุ๊กตา*" ประกอบการเล่ารายละเอียดต่างๆ ที่เกี่ยวกับการแบ่ง partitions ในโลกของ Linux ไปบ้างแล้ว ซึ่งต้องถือว่าเป็นการ "*เล่าย้อนศร*" ไปบ้าง เพราะโดยปกติแล้ว คนอื่นๆ เขาจะเรียนเรื่อง partitions นี้จากโปรแกรม fdisk มาก่อน ซึ่งมีรายละเอียดมากกว่า เวลามาเห็น Disk Druid ก็เลยไม่ต้องเล่าอะไรกันมาก เพราะว่าง่ายกว่ากันเยอะอยู่แล้ว ... หลายคนก็มองว่า Disk Druid จะมีโอกาสเห็นเพียงครั้งเดียวในตอนที่เราติดตั้งเท่านั้น และคิดว่ามันไม่น่าสนใจเท่ากับ fdisk ที่เรามีโอกาสได้ใช้งานบ่อยครั้งกว่าเท่าที่จำเป็น ... ก็เป็นมุมที่มองกันคนละแบบครับ ... เพราะผมอยากให้เห็นภาพรวมๆ ของมันก่อนที่จะเริ่มลงสู่รายละเอียดของคำสั่งหรือโปรแกรมที่เรียกว่า fdisk อันลือลั่นของ UNIX และ Linux

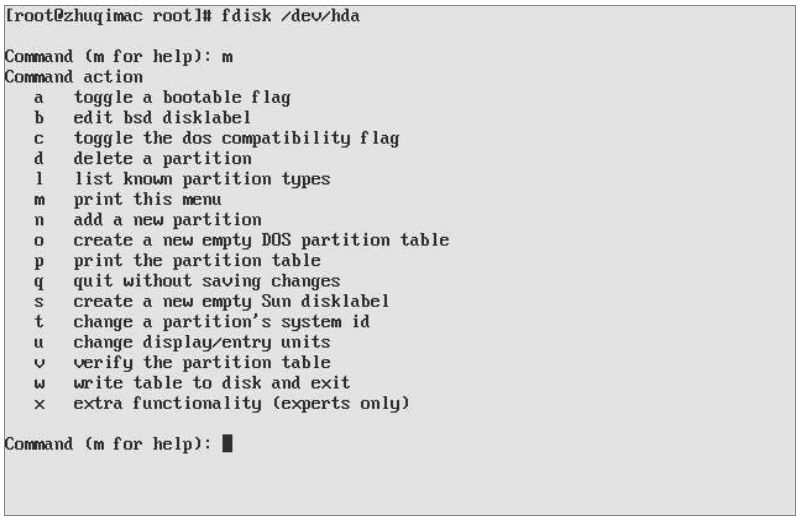
ทำความรู้จักกับ fdisk

ผมมีความเชื่อเสมอว่า การจัดการกับคอมพิวเตอร์อันดับแรกสุดนั้นก็คือ การจัดการกับ hard disk ของมัน เพราะนี่คือการกำหนดว่าเราจะใช้พื้นที่มากน้อยเท่าไหร่สำหรับการทำงานอะไรได้บ้างกับเครื่องมือชิ้นนั้นๆ ของเรา ... การยอมเสียเวลาเรียนรู้เกี่ยวกับคำสั่งที่นานปีทีหนถึงจะมีโอกาสได้ใช้งานจริงๆ นั้น อาจจะฟังดูเป็นเรื่องที่บ้าบอ ... แต่กว่า user คนหนึ่งๆ จะหาญกล้าเรียกใช้คำสั่ง fdisk นั้น ... ส่วนใหญ่แล้วก็มักจะเป็น user ที่ผ่านร้อนผ่านหนาวกับเครื่องคอมพิวเตอร์มานานพอสมควรทีเดียว ... ดังนั้นผมจึงกำหนดทฤษฎีขึ้นมาว่า **การเรียนรู้คำสั่ง fdisk ตั้งแต่แรกๆ นั้น จะเสมือนหนึ่งเป็น "*การเรียนกวดวิชา*" ในด้านการใช้งานคอมพิวเตอร์สำหรับ user มือใหม่ทุกคน** ... เพราะมันคือการเร่งให้ user ต้องผ่านร้อนผ่านหนาวกับรายละเอียดปลีกย่อยอีกหลายเรื่องในการจัดการกับคอมพิวเตอร์ด้วยตัวของเขาเอง

fdisk ถือเป็นหนึ่งใน "*คำสั่งอันตราย*" หรือ "*คำสาป*" ที่ระบบปฏิบัติการส่วนใหญ่จะซ่อนมันจากความซุกซนของ user ทั่วๆ ไป ... โดยวิธีการซ่อนของ MS-DOS และ MS-Windows เท่าที่เคยปฏิบัติกันมาก็คือ "*ไม่ยอมเอ่ยถึงมัน*" ทั้งๆ ที่มันยังถูกติดตั้งเข้าไปในระบบเหมือนปกติ ... แต่สำหรับ Linux แล้ว มันถูกกำหนดให้เป็นคำสั่งของ super-user หรือ root โดยเฉพาะ ด้วยการติดตั้งมันลงไปใน /sbin ซึ่งปกติแล้วจะมีแต่ root เท่านั้นที่สามารถเรียกใช้งานโดยตรง ...

เวลาเรียกใช้คำสั่ง fdisk ใน Linux จะต้องระบุชื่อ device หรือชื่อ hard disk ให้กับมันด้วยเสมอ เช่น fdisk /dev/hda ... แล้วเราก็จะเข้าสู่โปรแกรมที่หน้าตาน่ารักน่าชังที่สุดตัวหนึ่งของโลก

จะเห็นว่าเกือบจะไม่มีอะไรตอบสนองกลับมาให้เราเลยครับ นอกจากข้อความสั้นๆ ที่บอกให้เรารู้ว่าเราสามารถขอดูรายการคำสั่งใช้งานทั้งหมดได้ด้วยการกด m ซึ่งก็ย่อมาจากคำว่า menu นั่นเอง ... และเมื่อเรากดลงไปจริงๆ มันก็ช่วยให้เราเห็นตัวมันมากขึ้นอีกหน่อยเดียวเท่านั้นครับ ... คือมีรายการยาวๆ โผล่ออกมาให้เรารับรู้ว่ามี key อะไรให้เรากดได้บ้าง หลังจากนั้นเราก็ต้องเลือกกดแต่ละข้อตามที่เราต้องการใช้งาน เพราะ fdisk จะไม่ได้ช่วยเราอะไรมากไปกว่าที่เราสั่งงานมันแต่ละครั้งแต่ละเคาะเลย ... แบบนี้ไงครับที่เขาเรียกว่า interactive คือจะต้องมี "*ปฏิสัมพันธ์*" กันระหว่างผู้ใช้งานหรือ user กับเครื่องคอมพิวเตอร์อยู่ตลอดเวลาการใช้งาน :-)



ที่เห็นข้างบนนั่นก็คือหน้าจอหลังจากที่เรากด m เพื่อขอดู menu คำสั่งที่ fdisk มีมาให้ครับ รายละเอียดของคำสั่งต่างๆ ในเมนูคงไม่ต้องเล่าดีกว่า เพราะ user ควรจะได้ทดลองสัมผัสและศึกษาจากการมีปฏิสัมพันธ์กับ Linux ด้วยตัวเอง *อย่างน้อยที่สุดคำสั่ง fdisk ในโลกของ Linux ก็มีความปลอดภัยในระดับหนึ่ง เพราะการปู้ยี่ปู้ยำในระหว่างที่อยู่ในโปรแกรมนี้จะยังไม่มีผลใดๆ ในการทำลาย*

ข้อมูลของ hard disk เลย **จนกว่า user จะกด w เพื่อเขียนลงไปใน hard disk จริง ๆ เท่านั้น** ... *แล้วก็หมายความว่าหากเราออกจากโปรแกรม fdisk ด้วยการกด q เฉย ๆ ทุกอย่างบน hard disk ลูกเดิมของเราก็จะยังปลอดภัยจากการทำลายล้าง 100%* ... ตรงนี้ไม่เหมือนกับ OS ค่ายอื่นๆ นะครับ เพราะปกติของการเปลี่ยนแปลงใดๆ ในระหว่างที่อยู่ในโปรแกรม fdisk ของ MS-DOS หรือ MS-Windows จะเกิดผลทันทีทุกขั้นตอนเสมอ ... user จึงไม่มีโอกาสแก้ตัวใดๆ ทั้งสิ้น!!

แต่ที่อยากจะให้ดูก็ตรง menu ตัว t ที่อธิบายรายการไว้ว่า change a partition's system id นั่นแหละครับ ... ปัจจุบัน (คือขณะที่เรียบเรียงเอกสารชุดนี้) Linux มี system id สำหรับแต่ละ partition ให้เลือกมากถึงประมาณ 92 ชนิดด้วยกัน ...

```
 0  Empty           1c  Hidden Win95 FA 70  DiskSecure Mult bb  Boot Wizard hid
 1  FAT12           1e  Hidden Win95 FA 75  PC/IX           be  Solaris boot
 2  XENIX root      24  NEC DOS         80  Old Minix       c1  DRDOS/sec (FAT-
 3  XENIX usr       39  Plan 9          81  Minix / old Lin c4  DRDOS/sec (FAT-
 4  FAT16 <32M      3c  PartitionMagic  82  Linux swap      c6  DRDOS/sec (FAT-
 5  Extended        40  Venix 80286     83  Linux           c7  Syrinx
 6  FAT16           41  PPC PReP Boot   84  OS/2 hidden C:  da  Non-FS data
 7  HPFS/NTFS       42  SFS             85  Linux extended  db  CP/M / CTOS / .
 8  AIX             4d  QNX4.x          86  NTFS volume set de  Dell Utility
 9  AIX bootable    4e  QNX4.x 2nd part 87  NTFS volume set df  BootIt
 a  OS/2 Boot Manag 4f  QNX4.x 3rd part 8e  Linux LVM       e1  DOS access
 b  Win95 FAT32     50  OnTrack DM      93  Amoeba          e3  DOS R/O
 c  Win95 FAT32 (LB 51  OnTrack DM6 Aux 94  Amoeba BBT      e4  SpeedStor
 e  Win95 FAT16 (LB 52  CP/M            9f  BSD/OS          eb  BeOS fs
 f  Win95 Ext'd (LB 53  OnTrack DM6 Aux a0  IBM Thinkpad hi ee  EFI GPT
10  OPUS            54  OnTrackDM6      a5  FreeBSD         ef  EFI (FAT-12/16/
11  Hidden FAT12    55  EZ-Drive        a6  OpenBSD         f0  Linux/PA-RISC b
12  Compaq diagnost 56  Golden Bow      a7  NeXTSTEP        f1  SpeedStor
14  Hidden FAT16 <3 5c  Priam Edisk     a8  Darwin UFS      f4  SpeedStor
16  Hidden FAT16    61  SpeedStor       a9  NetBSD          f2  DOS secondary
17  Hidden HPFS/NTF 63  GNU HURD or Sys ab  Darwin boot     fd  Linux raid auto
18  AST SmartSleep  64  Novell Netware  b7  BSDI fs         fe  LANstep
1b  Hidden Win95 FA 65  Novell Netware  b8  BSDI swap       ff  BBT
Hex code (type L to list codes): 
```

ผมเองก็ไม่ชัดเจนว่า partition แต่ละ format ในโลกของคอมพิวเตอร์นั้นมันมีจุดเด่นจุดด้อยแตกต่างกันมากน้อยแค่ไหน แต่การที่ Linux ถูกออกแบบมาให้สามารถจัดการได้อย่างหลากหลายรูปแบบนั้นเป็นปัจจัยหนึ่งที่มันสามารถติดตั้งลงไปในเครื่องเดียวกันกับ OS ตัวอื่นๆ ได้อย่างค่อนข้างจะกลมกลืน ในขณะเดียวกันก็น่าจะเป็นปัจจัยหนึ่งที่ทำให้ Linux สามารถถูกใช้งานกับเครื่องคอมพิวเตอร์ได้หลากหลายที่สุดด้วย ไล่ไปตั้งแต่เครื่อง laptop, desktop, server ต่างๆ, mini-computer จนถึงระดับ mainframe ... ***มันคือประสิทธิภาพแอบแฝงที่นอกเหนือจากความ "ฟรี" ในด้านค่าลิขสิทธิ์ของมัน เพราะการใช้งาน Linux ในองค์กรธุรกิจจริง ๆ นั้น จะช่วยประหยัดบุคลากรด้าน IT ลงไปอย่างมหาศาลเลยทีเดียว เนื่องจากความจำเป็นที่จะต้องมีผู้เชี่ยวชาญเฉพาะทางของแต่ละ OS ในเครื่องคอมพิวเตอร์แต่ละ platform ก็จะหมดไปในทันทีด้วย ... เพราะ Linux คือหนึ่งเดียวที่สามารถติดตั้งได้ทุก platform ของ hardware ในปัจจุบัน และสามารถปฏิบัติงานได้ในหลากหลายหน้าที่ตามที่เราจะเป็นผู้กำหนดให้กับมันอย่างเฉพาะเจาะจง*** ... กระแสความตื่นตัวของ Linux จึงไม่ใช่เป็นเรื่องของแฟชั่นที่เห่อกันไปตามยุคตามสมัย แต่มันคือการปฏิวัติของวงการ IT ที่ครั้งหนึ่งเคยแตกกระสานซ่านเซ็นจนหาจุดร่วมที่แท้จริงไม่ได้เลย

*กลับมาที่เรื่องของ **fdisk** กันต่ออีกนิดหน่อย ...*

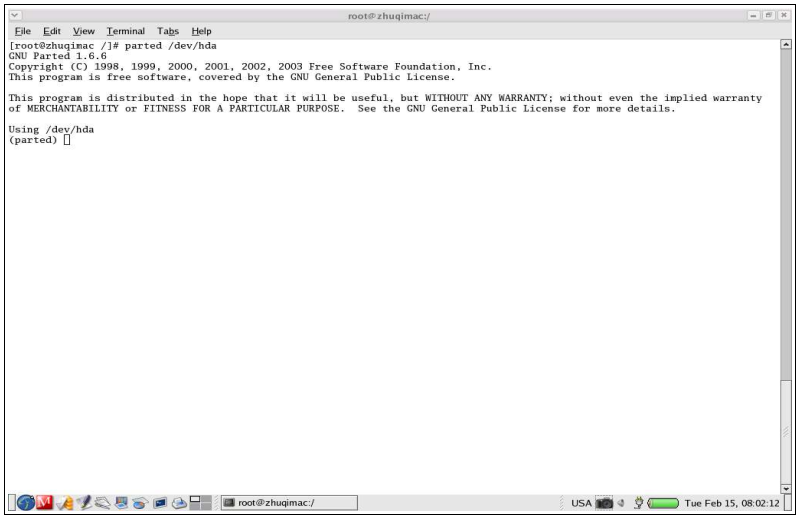
ความไม่สะดวกของ fdisk เมื่อเทียบกับ Disk Druid ก็คือ มันไม่มีคำสั่งในการสร้าง mount point เพื่อการติดตั้งระบบในคราวเดียวกัน ดังนั้นในระหว่างการติดตั้งระบบครั้งแรกของ Linux เกือบทุกค่ายในปัจจุบันนี้จึงได้ตัด option ของการแบ่ง partition ด้วย fdisk ออกไปหมดแล้ว (ยกเว้นค่ายของ slackware ที่ยังคงความ classic ของมันเอาไว้เสมอ :-)) ... โดย Disk Druid จะเป็นโปรแกรมที่ติดอยู่บนแผ่นที่ใช้ในการ install และจะถูกเรียกใช้งานเฉพาะเมื่อทำการติดตั้งเท่านั้น ส่วนโปรแกรมที่ใช้ในการจัดการเกี่ยวกับ partitions ต่างๆ ของ hard disk ที่จะถูกติดตั้งลงไปให้กับเรา เพื่อจะสามารถเรียกใช้งานได้ในโอกาสอื่นๆ ก็จะมีแค่ fdisk กับ partprobe และคำสั่งที่ทรงพลังกว่าอย่าง parted เท่านั้นเอง

นอกจาก fdisk จะไม่ได้รวม function ในการกำหนด mount point เอาไว้ด้วยแล้ว มันก็ยังมีข้อจำกัดเกี่ยวกับการใช้งานอีกนิดหน่อย นั่นก็คือ partitions ที่มีการเปลี่ยนแปลงใดๆ เนื่องจากคำสั่งใน fdisk ทั้งหมด แม้ว่า user จะสั่ง write ลงไปเรียบร้อยแล้วก็ตาม ระบบของเราก็จะยังไม่รับรู้ถึงการเปลี่ยนแปลงนั้นๆ จนกว่าเครื่องของเราจะได้รับการ restart ใหม่อีกครั้งหนึ่งก่อนเสมอ ... ความไม่สะดวกนี้เองที่ทำให้เกิด software ตัวเล็กๆ เข้ามาดับความรำคาญให้กับ user ... นั่นก็คือคำสั่ง partprobe ... ซึ่งโดยปกติหลังจากที่เราใช้คำสั่ง fdisk ในการจัดการอะไรบางอย่างเรียบร้อยแล้วนั้น เราก็จะเรียกโปรแกรม partprobe ให้ทำการอ่านค่า partitions ที่เกิดการเปลี่ยนแปลงนั้นอีกครั้งหนึ่ง แทนที่จะต้อง restart เครื่องให้วุ่นวายอารมณ์

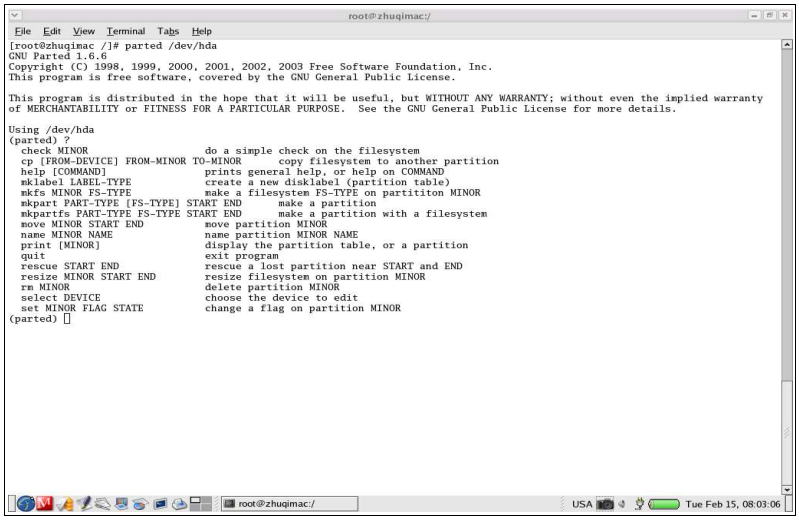
อย่างไรก็ตาม ในระหว่างการทำเอกสารชุดนี้ ผมได้มีโอกาสทดลอง Linux บนเครื่อง Macintosh ซึ่งถือเป็นเครื่องคอมพิวเตอร์ต่างตระกูลกับ PC ที่เราคุ้นหน้าคุ้นตากันอยู่ในภาคปกติ และได้เรียนรู้มาว่า คำสั่ง fdisk นั้นถูกออกแบบมาสำหรับเครื่องคอมพิวเตอร์ในกลุ่มที่ใช้ chip ของ intel เท่านั้น (ผมยังไม่มีโอกาสทดลองกับเครื่องที่ใช้ chip จากค่าย AMD ว่าเป็นยังไงเหมือนกัน) ... ก็ต้องถือว่าเป็นสถานการณ์ภาคบังคับที่ผมจะต้องทดลองใช้คำสั่ง parted เป็นครั้งแรก เพื่อจัดการกับ partitions ต่างๆ ให้กับเครื่อง Macintosh จอมดื้อที่ผมเอามาใช้ประกอบการศึกษา Linux และการเขียนเอกสารชุดนี้

เรื่องเล่าของคำสั่ง parted

ผมได้ยินชื่อคำสั่งนี้ครั้งแรกเมื่อตอนที่ไป take course ของ Red Hat นี่เอง ไม่ใช่คำสั่งที่น่าจะคุ้นเคยเลยครับ เพราะ fdisk เป็นคำสั่งที่แพร่หลายมากกว่า ไม่ว่าจะเป็น UNIX, Linux หรือแม้แต่ในโลกของ MS-DOS และ MS-Windows ... แม้ว่าคำสั่ง parted จะถูกติดตั้งคู่กับ Linux มานานพอสมควรแล้ว แต่ก็มีไม่กี่คนที่จะเอ่ยถึงมัน เพราะว่าหากเรารู้สึกว่าคำสั่ง fdisk มี interface ที่ไม่สะดวกกับการใช้งานแล้วล่ะก้อ ... คำสั่ง parted ก็จะเป็นสิ่งที่ยืนยันว่า "*นรกมีจริง*" สำหรับ user เลยทีเดียว ... แล้วนี่ก็คือหน้าจอเมื่อเราเรียกใช้คำสั่ง parted ล่ะครับ ...



คือถ้าเป็น fdisk ยังบอกนะว่าให้กด m เพื่อดู menu แต่เจ้า parted ไม่บอกอะไรเลย การเรียกดู menu จริงๆ จะต้องกดแป้น '?' หรือพิมพ์คำว่า help เท่านั้นครับ แล้วมันก็จะมีคำสั่งใช้งานทั้งหมดโผล่ออกมาให้ดูกัน แต่ก็มีแค่นั้นจริงๆ ครับ



ออกตัวไว้ซักหน่อยว่าหน้าจอที่ dump มานั้นเป็นคนละตัวกับภาพชุดแรกๆ เพราะหน้าจอที่ dump มาใหม่นี้เป็นหน้าจอของ YellowDog Linux ที่ run บนเครื่อง Macintosh 100% ไม่ใช่เป็นการสร้าง virtual machine ขึ้นมาทดลองเหมือนกับท่อนแรกๆ ของเอกสาร

สิ่งที่น่าสนใจของ parted ก็คือ มันสามารถ copy แบบยก partition กันทั้งยวงได้เลย ไม่ใช่สร้าง partition ขึ้นมาแล้วค่อยมาสั่ง copy กันทีหลัง แต่ parted จะทำงานเหมือนกับ dump ข้อมูลกันทั้ง partition ในคราวเดียวไปเลย แล้วมันก็ยังสามารถปรับเปลี่ยนขนาดของ partition ต่างๆ ได้ด้วย โดยไม่มีการทำลายข้อมูลเดิมที่มีอยู่ใน partition นั้นๆ ... ซึ่งในโลกของ MS-Windows จำเป็นต้อง

อาศัย software ประเภท third party ที่ชื่อว่า Partition Magic มาจัดการหน้าที่ตรงนี้แทน ...

ลำพังแค่ที่เห็นตรงหน้าจอนั้นแทบจะไม่ได้ช่วยให้รู้อะไรเลยครับสำหรับคำสั่ง parted นี้ ต่างกับคำสั่ง fdisk ที่แม้ว่าจะไม่สวยงามแต่ก็สื่อสารกับ user ได้ดีที่ระดับหนึ่ง ***เพราะฉะนั้นผมก็เลยจัดการ download ไฟล์คู่มือของ parted มาเก็บเอาไว้ซะเลย*** เพราะมันคือคำสั่งที่ทรงอานุภาพมากในการจัดการกับ hard disk ของ Linux ... แต่ในขณะเดียวกันมันก็มีอำนาจการทำลายล้างสูงมากด้วย เพราะ parted จะเขียนข้อมูลของ partition ที่ถูกกำหนดใหม่ลงไปสู่ hard disk ทันที ... แน่นอนที่เราสะดวกสบายกับการไม่ต้อง restart เครื่องหรือเรียกใช้คำสั่ง partprobe อีก แต่เราก็อาจจะต้อง re-install ทุกอย่างใหม่อีกรอบก็ได้ถ้าขืนทำอะไรสุ่มสี่สุ่มห้าลงไปกับคำสั่งประเภทนี้ ... สำหรับข้อความทั้งหมดในคู่มือฉบับสมบูรณ์ของ parted ผมไม่ได้เอามารวมไว้ในเอกสารฉบับนี้ เพราะมันยาวประมาณ 70-80 หน้ากระดาษ ใครที่สนใจจะอ่านก็ช่วยเข้าไปหาอ่านกันเอาเองใน server ของบริษัทฯ ก็แล้วกันครับ :-)

...

เอาล่ะ ... ที่เล่ามาซะยืดยาวนั่นก็คือส่วนที่ผมถือว่าเป็น "*ความรู้พื้นฐาน*" เกี่ยวกับ Linux ครับ เพราะว่าพอเราเรียนรู้เรื่องของ partition ซึ่งก็คือเรื่องของการจัดการกับ hard disk ไปแล้ว เราก็จะเริ่มทำการติดตั้งโปรแกรมหรือที่เรียกว่า install หรือ setup นั่นแหละ แล้วถึงจะเริ่มต้นใช้งานกัน ... ผมจะเว้นส่วนที่เป็นเรื่องของการติดตั้งออกไป เพราะ Linux ต่างค่ายต่างสำนักก็จะมีวิธีการติดตั้งและหน้าจอที่สื่อสารกับ user ที่แตกต่างกันออกไป ช่วยไปหาอ่านเอาเองจากคู่มือของค่ายที่เราเลือกใช้ก็แล้วกัน ซึ่งส่วนใหญ่ก็จะเป็น graphic กันไปหมดแล้ว ไม่ได้โหดร้ายทารุณเหมือนกับ Linux ยุคแรกๆ อีกแล้วล่ะครับ ... โดยภาพรวมๆ ของ Linux หลายๆ ค่ายเท่าที่เคยสัมผัสมาแล้วนั้น ผมว่ามันดูเข้าใจได้ง่ายกว่าของ MS-Windows ซะด้วยซ้ำในเวลานี้ :-)

ดังนั้น ผมจึงขออนุญาตที่จะกระโดดข้ามไปหัวข้อใหม่กันเลยก็คือเรื่องของ Bootup Sequence หรือลำดับขั้นตอนในการ boot เข้าสู่ระบบของ Linux ซึ่งเป็นตอนที่ต่อจากการติดตั้งที่ผมขอเว้นข้ามไปนั่นเอง ... มันเป็นขั้นตอนการทำงานที่มีความสวยงามของกระบวนการคิดและมีความเป็นระเบียบเรียบร้อยของ Linux ที่พัฒนาต่อยอดออกมาจาก UNIX ซึ่งผมเชื่อว่าเป็นสุดยอดแห่งการพัฒนาระบบปฏิบัติการเลยทีเดียว !!

ส่วนที่ 2 : Bootup Sequence

โดยปกติ หลังจากที่เรากด switch เพื่อเปิดเครื่องแล้ว เราก็จะเดินไปขี้หรือไปเยี่ยวอย่างไม่ค่อยจะสนใจอะไรกับเครื่องคอมพิวเตอร์ของเราเองซักเท่าไหร่ เพราะมันยังไม่พร้อมที่จะให้เราเข้าไปแทรกแซงอะไรในระหว่างการ boot ของมันเลย ... ซ้ำร้ายการแทรกแซงอย่างผิดกาละเทศะในบางกรณีก็อาจจะทำให้ระบบพังไปเลยก็ไม่แน่เหมือนกัน :-)

สมัยหนึ่งในโลกของ MS DOS เครื่องคอมพิวเตอร์ของเราก็จะทำงานเงียบๆ อยู่พักเดียว แล้วถึงจะเริ่มทำงานจากชุดคำสั่งใน AUTOEXEC.BAT ที่หลายคนเชื่อว่ามันคือ "*หนึ่งเดียว*" ที่ทำงานหลังจากการ boot ระบบ (ซึ่งจริงๆ แล้วเราสามารถที่จะสั่งให้ MS DOS ไปทำงานที่ไฟล์ชื่ออะไรก็ได้ในขั้นตอนของการ boot ... เพียงแต่ไม่ค่อยจะมีคน "*มือคัน*" ในเรื่องนี้เท่านั้นเอง) ... ในระยะต่อมาที่โลกของอุปกรณ์คอมพิวเตอร์มีความหลากหลายและซับซ้อนมากขึ้น ชุดคำสั่งที่ต้องประกอบการ boot ก็เลยมากขึ้นไปด้วย ระยะเวลาในการ boot ก็ใช้เวลานานกว่าเดิมเยอะทีเดียว ... ทุกๆ OS ก็เลยพยายามสร้าง interface บางอย่างขึ้นมาเพื่อที่จะสื่อสารกับ user ว่า ... "*ฉันยังมีลมหายใจและยังทำงานอย่างขมักเขม้นอยู่นะ ... กรุณาอดทนรอให้ถึงที่สุดจนกว่าฉันจะ boot ตัวเองให้เรียบร้อยและกรุณาอย่าเสือก*" ... ผมประมาณเอาว่าแถบสีวิ่งๆ บนจอของ MS Windows หรือของ MacOS กำลังพยายามสื่อสารข้อความที่ว่านั่น ... แต่สำหรับ Linux มักจะนิยมแสดง interface แบบข้อความเพื่อสื่อสารว่าทำงานส่วนไหนเรียบร้อยแล้วหรือทำไม่สำเร็จตรงส่วนไหนบ้าง ... เป็น interface ที่บางคนเริ่มจะเหม็นเบื่อแล้ว และเริ่มจะมีการสร้าง interface แบบ quiet boot ที่เห็นแต่แถบสีหยาบๆ คายๆ นั้นขึ้นมาแทน ... แต่เกือบทุก OS ในสมัยนี้ก็ต้องใช้เวลาประมาณ 1 เยี่ยวกว่าจะจบขั้นตอนแรกนี้ของพวกมัน ... ซึ่งความรู้สึกที่ว่า "*สนใจได้แต่ห้ามเสือก*" กับหน้าจอที่ด้อยประสิทธิภาพในการสื่อสารของ OS หลายๆ ค่ายนี้เองที่ทำให้ user เกือบทั้งโลกขี้เกียจจะรับรู้อะไรๆ ในระหว่างการ boot ระบบของเครื่องมือชิ้นที่เรียกกันว่าคอมพิวเตอร์นี้

ส่วนหนึ่งของอาการ "*ละเลยความรู้*" ของ user ก็เกิดมาจากการที่ OS เกือบทั้งโลกพยายามเก็บงำการทำงานของตัวเองให้เป็น "*ความลับที่ซับซ้อน*" ซึ่งแตกต่างจากระบบของ Linux ที่พยายามสร้างระบบให้ "*เปิดเผยอย่างซับซ้อน*" แทน ... เพราะเราต้องอย่าลืมนะว่า ต้นกำเนิดจริงๆ ของ Linux นั้นเกิดจากแวดวงของการศึกษา และต้องการที่จะเผยแพร่ความรู้ให้กับนักเรียนนักศึกษาเป็นหลัก มันจึงเป็นทัศนคติและมุมมองที่แตกต่างจาก OS เชิงพาณิชย์ค่ายอื่นๆ มาตั้งแต่แรก ... การรู้จักเก็บเกี่ยวความรู้จากขั้นตอนนี้จึงเป็นประโยชน์อย่างมากต่อความเข้าใจในระบบการทำงานของเครื่องมือที่เราเลือกใช้ ซึ่งความเข้าใจนั้นๆ ก็จะเป็นประโยชน์ต่อการดูแลรักษาและปรับแต่งคุณสมบัติต่างๆ ของเครื่องมือนี้ด้วยตัวของเราเองต่อไปได้อย่างอิสระ ... ความแข็งแรงและความปลอดภัยของระบบ Linux ส่วนใหญ่ก็เกิดมาจาก "*การเปิดเผยความลับ*" ส่วนนี้ออกมาให้แก่สาธารณชนรับรู้ ซึ่งส่งผลให้มีการปรับแต่งระบบอย่างเฉพาะเจาะจงที่หลากหลายจนยากแก่การโจมตีระบบด้วยวิธีใดวิธีหนึ่งอย่างตายตัว

... ในความเห็นส่วนตัวของผมแล้ว ผมเชื่อว่า Linux มีความเรียบง่ายในตัวของมันเองเป็นอย่างมาก ไม่ได้สลับซับซ้อนเหมือนกับ OS ค่ายอื่นๆ อย่างที่เราหลงเชื่อ แต่มันเป็นความเรียบง่ายที่เราจำเป็นต้องมีความรู้ไปจัดการกับมัน เพราะมันเกือบจะส่งต่อมาให้เราแบบดิบๆ โดยที่เราสามารถปรุงแต่งของเราเองต่อไปได้ ขนาดของระบบ Linux ทั้งระบบจึงเล็กกระทัดรัด ไม่ได้ใหญ่โตมโหฬารเหมือนกับ OS ตัวอื่นๆ ... มันคล้ายกับของเล่นที่ส่งต่อมาให้เราเป็นเพียงชิ้นส่วนย่อยๆ เพื่อให้เราเอามาประกอบเป็นรูปเป็นร่างต่างๆ กันเอาเอง ในขณะที่ OS ตัวอื่นๆ จะหล่อเป็นรูปสำเร็จที่พร้อมใช้พร้อมเล่นอย่างจำกัดจำเขี่ยเท่าที่มันถูกออกแบบเอาไว้อย่างสลับซับซ้อน ... อิฐที่เห็นเป็นก้อนๆ ยังไงก็ซับซ้อนน้อยกว่าอาคารที่สร้างสำเร็จรูปอยู่แล้ว โดยส่วนที่แตกต่างกันก็คือ เรายังสามารถใช้อิฐเหล่านั้นมาประกอบ

เป็นสิ่งปลูกสร้างอย่างอื่นหรือรูปร่างหน้าตาแบบอื่นได้อย่างอิสระ ในขณะที่เราจะไม่สามารถเปลี่ยนแปลงความสลับซับซ้อนที่สำเร็จด้วยมือคนอื่นไปก่อนแล้วเท่านั้นเอง ... การต้องปรุงอาหารด้วยตัวเองอาจจะสลับซับซ้อนกว่าการสั่งซื้ออาหารสำเร็จมาแดกๆ กันเข้าไปให้หมดเรื่อง แต่มันเป็นความสลับซับซ้อนในระดับของขั้นตอนหรือกระบวนการ ไม่ใช่ความสับซับซ้อนที่ตัวเครื่องปรุงและวัตถุดิบ ... ในมุมมองแบบนี้ Linux ก็ยังไม่ถึงกับป่าเถื่อนขนาดส่งต่อมาให้เราเป็นแค่ดินกับถุงปุ๋ยแล้วก็เมล็ดพันธุ์ให้ไปปลูกผักปลูกหญ้าสำหรับกินกันเอง แต่มันถูกจัดเตรียมไว้ให้แล้วในระดับหนึ่ง พร้อมที่จะใช้ พร้อมที่จะเล่น แต่ก็พร้อมที่จะปรุงเสริมเพิ่มรสอย่างที่เราต้องการได้เสมอ ... อย่างนี้เห็นภาพชัดมั้ย?!

เอาล่ะ ... มาดูกันซิว่า Linux ทำอะไรมั่งในขณะที่มันกำลัง boot ตัวเองขึ้นมา :-)

ภาพรวมๆ ของขั้นตอนในการ boot ระบบของ Linux แบ่งออกเป็น 4 ขั้นตอนสั้นๆ แค่นี้คือ

1. BIOS initialization
2. Boot Loader
3. Kernel initialization
4. init starts and enters desired run level by executing:
   - /etc/rc.d/rc.sysinit
   - /etc/rc.d/rc and /etc/rc.d/rc?.d
   - /etc/rc.d/rc.local
   - X Display Manager if appropriate

1. BIOS initialization : การเริ่มต้นระบบของ BIOS

BIOS (ย่อมาจาก Basic Input/Out System) คือชุดคำสั่งพื้นฐานที่ถูกจัดเก็บไว้ใน ROM สำหรับให้ระบบปฏิบัติการหรือ OS หนึ่งๆ เรียกใช้เพื่อการควบคุมและสั่งการชิ้นส่วนหรืออุปกรณ์ต่อพ่วงอื่นๆ ของเครื่องคอมพิวเตอร์นั้นๆ ... ปกติแล้ว BIOS จะเริ่มทำงานด้วยตัวของมันเองหลังจากที่เรา start เครื่อง ไม่ว่าจะโดยการเปิดสวิทช์ขึ้นมาใหม่หรือทำการ restart ด้วยวิธีการใดๆ ก็ตาม

สิ่งแรกที่ BIOS ทำก็คือ POST หรือ Power On Self Test เพื่อตรวจสอบความพร้อมของอุปกรณ์ต่างๆ ของมันจากทะเบียนข้อมูลที่จัดเก็บไว้ใน CMOS (Complementary Metal Oxide Semiconductor) โดย CMOS นี้ได้รับการหล่อเลี้ยงหน่วยความจำของมันด้วย battery เล็กๆ บนแผงวงจร main board อยู่ตลอดเวลา เพื่อให้ BIOS สามารถอ่านค่าได้ว่า ในระบบของ hardware ประจำเครื่องนั้นๆ มีอะไรประกอบอยู่บ้าง เช่นมี hard drive กี่ตัว, มี floppy drive หรือไม่, ติดตั้ง CD-ROM หรือ DVD-ROM อยู่บ้างรึเปล่า หรือว่ามี Network Interfaced Card กี่ชิ้นกี่แผง แล้วสามารถ boot ระบบจากส่วนไหนของ hardware ที่มีอยู่? เรียงลำดับกันยังไง? ตลอดไปจนถึงวันที่, เวลา, และรายละเอียดปลีกย่อยอื่นๆ ของ hardware ที่ประกอบอยู่นั้น

พอทำงานในส่วนของ POST เรียบร้อยแล้ว BIOS ก็จะพยายามอ่านค่าของ boot sector จาก boot devices หรืออุปกรณ์ต่างๆ ที่จะทำการ boot ระบบตามลำดับรายการของมัน เช่นจาก floppy drive, hard drive, CD-ROM drive, หรืออาจจะเป็น Network Interfaced Card ขึ้นอยู่กับว่าเราจะสั่งให้มันเรียงลำดับก่อนหลังยังไงด้วย ... boot sector นี้ก็คือพื้นที่ส่วนต้นๆ ที่มีขนาดเพียง 512 bytes ของ media ที่เราเลือกไม่ว่าจะเป็น floppy disk, hard disk, หรือ CD-ROM ... เมื่อ BIOS เจอค่าของ boot sector จาก media ไหนก่อน ก็จะทำการ boot เข้าสู่ระบบด้วย boot device นั้นๆ

การทำงานในส่วนของ BIOS นี้ยังไม่เกี่ยวกับระบบปฏิบัติการใดๆ เลย เพราะเป็นการปฏิบัติตามคำสั่งที่ฝังมากับ ROM (Read Only Memory) หรือ Firmware ของเครื่องคอมพิวเตอร์เท่านั้น

2. Boot Loader

ทันทีที่ BIOS ทำงานของมันเองเสร็จสมบูรณ์แล้ว มันก็จะอ่านค่าของ boot sector จาก media ที่ถูกเลือก หน้าที่ในการดูแลระบบและการทำงานต่อจากนั้นก็ถูกโอนถ่ายไปให้กับ Boot Loader ของระบบปฏิบัติการหรือ OS ที่เราเลือกใช้

ในระบบปฏิบัติการรุ่นโบร่ำโบราณหน่อยอย่างเช่น MS-DOS, MS Windows 98 หรือ MS Windows ME อาจจะไม่มี Boot Loader ให้เรียกใช้งานกันครับ แต่ใน Linux จะมี Boot Loader มาให้ใช้งานกันตั้งแต่ยุคแรกๆ คือ LILO (**LI**nux **LO**ader) และพัฒนาเพิ่มเติมขึ้นมาอีกตัวหนึ่งคือ GRUB (**GR**and **U**nified **B**ootloader)

ผมบังเอิญได้มีโอกาสเห็น LILO ในช่วงท้ายๆ ของอายุขัยของมันจากการติดตั้ง Slackware v8.1 หน้าตาน่าเกลียดแบบ text mode แท้ๆ เลยครับ ซึ่งก็จะไม่ค่อยได้เห็นกันอีกแล้ว เพราะ Linux รุ่นใหม่ๆ จะกำหนดให้ GRUB เป็น Boot Loader มาตรฐานแทน ... แต่ LILO จะเหมาะสำหรับเครื่องรุ่นเก่าๆ ที่เรายังนึกเสียดายแล้วอยากจะมีไว้ใช้งานแบบ text mode อยู่น่ะครับ :-) ... มันก็เลยทำให้ระยะหลังๆ นี้ผมเรียนรู้เฉพาะเรื่องของ GRUB เพียงตัวเดียวเท่านั้น

ปกติของ Boot Loader จะแยกออกเป็น 2 ท่อน ท่อนแรกเรียกว่า Initial Program Loader (IPL) เป็นตัวที่รับหน้าที่ต่อมาจาก BIOS โดยมักจะมีขนาดเล็กๆ เพียง 446 bytes เท่านั้น แล้วก็มักจะได้รับการติดตั้งอยู่ใน MBR (Master Boot Record) ของ media ที่เราเลือกใช้เสมอ มันจะทำหน้าที่แค่จดจำตำแหน่งที่เก็บของ Boot Loader ท่อนที่ 2 ของมัน หรืออาจจะเป็นท่อนแรกของระบบปฏิบัติการอื่นๆ ที่จะถูก boot ขึ้นมาแทน ... GRUB ก็คือเจ้าท่อนแรกที่ว่านี้ โดยการทำงานของมันจะถูกควบคุมผ่านไฟล์เล็กๆ ตัวหนึ่งที่ชื่อว่า /boot/grub/grub.conf

แล้วนี่ก็คือตัวอย่างของ /boot/grub/grub.conf ของเครื่องที่ผมใช้เพื่อเขียนเอกสารชุดนี้ครับ

```
default=0
timeout=30
splashimage=(hd0,2)/grub/splash.xpm.gz
# hiddenmenu
title Microsoft Windows 98 SE/Thai
        rootnoverify (hd0,0)
        chainloader +1
title Fedora Core (2.6.10-1.766_FC3)
        root (hd0,2)
        kernel /vmlinuz-2.6.10-1.766_FC3 ro root=LABEL=/ rhgb quiet
        initrd /initrd-2.6.10-1.766_FC3.img
title Fedora Core (2.6.9-1.667)
        root (hd0,2)
        kernel /vmlinuz-2.6.9-1.667 ro root=LABEL=/ rhgb quiet
        initrd /initrd-2.6.9-1.667.img
```

มันเป็น text file ธรรมดาที่เราสามารถแก้ไขได้ไม่ยากเย็นนัก คล้ายๆ กับ boot menu ใน config.sys รุ่นหลังๆ ของ MS-DOS อย่างนั้นแหละ แต่สำหรับ grub.conf จะมีรายการบันทึกไว้แค่ตำแหน่งแห่งที่เก็บของ kernel และ boot image หรือ Boot Loader ท่อนที่สองของระบบเท่านั้นเอง เมื่อเราเลือกเมนูตัวใดตัวหนึ่งแล้ว GRUB ก็จะจัดการเรียก kernel และ boot image นั้นๆ ขึ้นมารับหน้าที่ต่อจากมันไป ... ในกรณีของเครื่องที่ใช้กับงานเอกสารชิ้นนี้ ถ้าเราเลือกท่อนที่เป็น Microsoft Windows 98 SE/Thai ระบบก็จะ boot เข้าสู่โลกของ Windows แทนที่จะเป็น Linux ... ส่วนในกรณีของ Linux มันก็ยังมีให้เลือก 2 version ด้วยกันเพราะมีการ update ระบบ

หลังจากการติดตั้งครั้งแรกไปแล้วนั่นเอง ... ตอนที่เราเลือก Fedora Core ซึ่งก็คือ Linux จากค่าย Red Hat นั่นเองที่ Linux ตัวจริงๆ หรือ kernel จะเริ่มต้นการทำงานของมัน และเข้าทำหน้าที่ควบคุมทุกๆ อย่างของ hardware ในระบบของมันต่อไป (Fedora เป็นชื่อ project ของ Red Hat หลังจากที่ Red Hat ปรับเปลี่ยนรูปแบบการดำเนินงานเป็น commercial มากขึ้น โดยแยก product ออกเป็น 2 lines อย่างชัดเจนคือ Red Hat เป็น Linux ที่จะดำเนินงานแบบ commercial ในขณะที่ Fedora จะเป็น Linux ที่ดำเนินงานแบบ community ต่อไป โดยคำว่า Fedora ก็คือคำศัพท์เฉพาะของ "*หมวกทรงปีก*" ที่ Red Hat ใช้เป็น logo ของตัวเองนั่นเอง)

รายละเอียดของ grub.conf มีเยอะพอสมควรครับทั้งๆ ที่มันมีหน้าที่แค่นี้เอง คนที่สนใจอยากจะอ่านกันจริงๆ ก็คงต้องติดตั้ง Linux ให้เรียบร้อยก่อน จากนั้นก็ออกไปที่ terminal screen หรือ command line mode แล้วพิมพ์คำสั่งว่า info grub เพื่อดูคู่มือฉบับเต็มของมันเอาเอง ... ขืนเล่ารายละเอียดในเอกสารฉบับนี้ มันคงจะดู'เวอร์'ไปหน่อยล่ะครับ :-)

3. Kernel initialization

ขั้นตอนนี้ถึงจะเริ่มต้นเข้าสู่ระบบปฏิบัติการหรือ OS กันจริงๆ ... kernel คือส่วนที่เป็นหัวใจการทำงานของระบบปฏิบัติการทุกตัว โดยทั่วไปแล้วมันจะมีหน้าที่ทำความรู้จักกับชิ้นส่วนและอุปกรณ์ต่อพ่วงต่างๆ ที่ต่อติดอยู่กับตัวเครื่องโดยการอ่านค่าจาก drivers ที่มันมีอยู่เพื่อใช้ในการติดต่อสื่อสารและควบคุมบังคับการทำงานของอุปกรณ์แต่ละชิ้นเหล่านั้น ... โดยปกติแล้วขั้นตอนของ kernel initialization จะใช้เวลาแป๊บเดียว และ Linux จะจัดการบันทึกผลลัพธ์ของการทำงานในขั้นตอนนี้ไว้ใน log file ตัวหนึ่งที่ชื่อว่า /var/log/dmesg ... เพื่อให้เราเอาไว้ตรวจดูว่า kernel ที่เราติดตั้งไว้นั้นมันทำความรู้จักกับอุปกรณ์ของเราครบถ้วนเรียบร้อยขนาดไหน

เมื่อ drivers ที่ระบบต้องการทั้งหมดถูก load ขึ้นมาเรียบร้อยแล้ว kernel ก็จะทำการเชื่อมโยง file systems ต่างๆ เพื่อเตรียมพร้อมสำหรับการใช้งาน เราเรียกขั้นตอนนี้ว่าการ mount file systems โดยเฉพาะ root file system (/) จากนั้น Linux ก็จะเรียกใช้งาน process แรกของระบบขึ้นมา (init) แล้วส่งผ่านการควบคุมระบบทั้งหมดไปที่ process ที่ชื่อว่า init นี้ต่อไป

4. init initialization

จะว่าไปแล้ว init ก็จะคล้ายๆ กับ AUTOEXEC.BAT ในระบบปฏิบัติการของ MS-DOS หรืออาจจะทำหน้าที่ใกล้เคียงกับ system.ini และ win.ini ในระบบปฏิบัติการของ MS-Windows อยู่บ้าง แต่รายละเอียดของ init จะมีมากมายกว่านั้น เพราะมันเล่นพ่วงเอาหน้าที่ที่คล้ายๆ กับ login script ในระบบ network เข้าไปอยู่ในตัวของมันด้วย ... ส่วนหนึ่งก็เพราะ Linux ถูกออกแบบให้เป็นระบบปฏิบัติการแบบ multi-user มาตั้งแต่เริ่มต้นชีวิตของมันนั่นเอง

การทำงานของ Linux ทั้งระบบเกือบจะเรียกได้ว่าทำงานตาม script ที่เขียนสั่งเอาไว้ล่วงหน้า โดย user สามารถที่จะเข้าไปแก้ไขดัดแปลงหรือเพิ่มเติมลูกเล่นต่างๆ ได้ด้วยตัวเองตลอดเวลา (ถ้ารู้ตำแหน่งที่จัดเก็บไฟล์เหล่านั้นทั้งหมดนะ :-)) ... ความสามารถอันล้นหลามของ Linux จึงขึ้นตรงกับผู้ดูแลระบบนั้นๆ เป็นหลัก โดย Linux จัดเตรียม script มาตรฐานมาให้แล้วจำนวนหนึ่งสำหรับพวก "*มือไม่คัน*" สามารถปฏิบัติงานไปได้อย่างราบรื่นพอสมควร แต่ก็เปิดช่องว่างขนาดมหึมาให้กับพวก "*มือคัน*" ได้มีโอกาสทดสอบหรือประลองความบ้าระห่ำของตัวเองด้วยเหมือนกัน ... เพราะ Linux เติบโตขึ้นมาใน "*โลกแห่งการเรียนรู้ตลอดเวลา*" นั่นเอง

init คือ "process แรก" และถือเป็น "process ราก" ของ Linux ทั้งระบบ โดยทุกๆ process หรือคำสั่งในการทำงานขั้นตอนอื่นๆ ทั้งหมดที่จะเกิดขึ้นต่อๆ ไปนั้น จะเป็นการแตกแขนงออกมาโดยมีจุดเริ่มต้นจาก init ตัวนี้เสมอ ... โดยปกติ init จะเป็น executable file ที่เราไม่สามารถอ่านด้วยภาษามนุษย์ทั่วๆ ไป แต่มันได้รับการออกแบบให้ถูกควบคุมขั้นตอนการทำงานทุกอย่างผ่าน script

ตัวหนึ่งที่มีชื่อว่า /etc/inittab ... เนื้อในของไฟล์ /etc/inittab เป็นตัวอักษรในระบบ ASCII หรืออักษรโรมันที่เราสามารถอ่านออกอย่างภาษาอังกฤษทั่วไป แต่จะมีโครงสร้างภาษาเฉพาะตัวที่เรียกว่า script เพื่อเรียกใช้งานโปรแกรมอื่นๆ อีกทอดหนึ่งในระบบของ Linux ... ซึ่งพอเราแกะเข้าไปดูกันจริงๆ แล้ว เราก็จะพบว่า inittab นั้นน่าจะย่อมาจากคำเต็มๆ ว่า **INIT**ialization **TAB**le นั่นเอง !!

ภายใน /etc/inittab มีระบุคำสั่งเพียงไม่กี่บรรทัด และโดยมากก็เป็นการกำหนดค่าตัวแปรให้กับ script ตัวอื่นไปทำงานต่อ ... ตัวแปรหลักๆ ที่กำหนดไว้ใน /etc/inittab ก็คือ run level ซึ่งปัจจุบัน Linux ออกแบบไว้เพียง 6 (คือ run level 0 ถึง run level 6 แต่ก็ยังไม่มีการกำหนดหน้าที่ใดๆ ให้กับ run level 4 เท่านั้น)

```
#
# inittab       This file describes how the INIT process should set up
#               the system in a certain run-level.
#
# Author:       Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
#               Modified for RHS Linux by Marc Ewing and Donnie Barnes
#

# Default runlevel. The runlevels used by RHS are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# When our UPS tells us power has failed, assume we have a few minutes
# of power left.  Schedule a shutdown for 2 minutes from now.
# This does, of course, assume you have powerd installed and your
# UPS connected and working correctly.
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"

# If power was restored before the shutdown kicked in, cancel it.
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"


# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
x:5:respawn:/etc/X11/prefdm -nodaemon
```

ไฟล์ /etc/inittab ถือว่ามีความสำคัญมากต่อการ boot ระบบของ Linux เพราะหากไฟล์นี้เสียหาย หรือกำหนดค่าบางอย่างผิดพลาด มันก็จะส่งผลให้ Linux ไม่สามารถ boot ตัวเองขึ้นมาได้เลย เรามาลองดูบรรทัดที่ป้ายแถบสีเหลืองๆ ไว้นั่น

```
id:5:initdefault:
```

ตรงนี้คือสั่งให้ Linux ใช้ run level 5 เป็นค่ามาตรฐานในการ boot ระบบ ซึ่งหากเรากำหนดค่าผิดไปเป็น run level 0 ระบบก็จะถือว่าการ boot ปกติคือการปิดเครื่องเสมอ หรือถ้ากำหนดค่าลงไปเป็น run level 6 ระบบก็จะวนเวียนเปิดๆ ปิดๆ ตัวเองตลอดเวลา เพราะ run level 6 หมายถึงการ reboot เครื่องคอมพิวเตอร์ของเรานั่นเอง

run level แต่ละตัวจะใช้คำสั่งในการ boot ที่แตกต่างกัน แล้วก็อาจจะไม่เท่ากันด้วย โดยตัวที่ควบคุม script ที่จะทำการ boot ระบบต่อจาก inittab ก็คือ /etc/rc.d/rc.sysinit ซึ่งจะมีการส่งต่อคำสั่งไปยัง script ที่ควบคุมแต่ละ run level ต่อไปอีกทอดหนึ่ง ... script ที่เกี่ยวกับการ boot ทั้งหมดนั้น Linux ได้จัดการเก็บรวบรวมไว้ในที่เดียวกันคือ /etc/rc.d แล้วแยกย่อยออกไปเป็น sub-directory เพื่อความเป็นระเบียบเรียบร้อยและเหมาะแก่การจัดการแต่ละ run level ต่อไป ... ถ้าดูจากตัวอย่างของ /etc/inittab ที่ลอกมาให้ข้างบนนั้น ก็จะเห็นว่า init จะเรียกใช้คำสั่งที่เกี่ยวกับ run level 5 ในการ boot ตามปกติของมัน (คำสั่งที่มีระบายสีฟ้าไว้) ... โดยทุกๆ run level จะถูกควบคุมด้วย script เพียง 2 ตัวเท่านั้นคือ /etc/rc.d/rc.sysinit และ /etc/rc.d/rc ที่ใช้เพียงตัวเลขของ run level นั้นๆ เป็นตัวแปรในการเรียกชุดคำสั่งที่ต้องการขึ้นมาใช้งาน

ขออนุญาตยกตัวอย่างของ run level 5 ซึ่งเป็น run level ที่เรียกใช้ Linux แบบ graphic mode อันเป็น interface ที่นิยมแพร่หลายอยู่ในปัจจุบัน ... run level 5 นั้นจะมีชุดของ script ที่ควบคุมอีกกระบุงใหญ่ที่ถูกจัดเก็บเอาไว้ใน /etc/rc.d/rc5.d ครับ

```
K01yum                  K02NetworkManager       K05saslauthd
K10psacct               K20nfs                  K24irda
K28amd                  K30spamassassin         K35vncserver
K35winbind              K36lisa                 K50netdump
K50snmpd                K50snmptrapd            K73ypbind
K74nscd                 K74ntpd                 K85mdmpd
K89named                K89netplugd             K90bluetooth
K94diskdump             K99microcode_ctl        S01sysstat
S04readahead_early      S05kudzu                S06cpuspeed
S08iptables             S09isdn                 S09pcmcia
S10network              S12syslog               S13irqbalance
S13portmap              S14nfslock              S15mdmonitor
S18rpcgssd              S19rpcidmapd            S19rpcsvcgssd
S25netfs                S26apmd                 S26lm_sensors
S28autofs               S33nifd                 S34mDNSResponder
S40smartd               S44acpid                S54hpoj
S55cups                 S55sshd                 S56xinetd
S80sendmail             S85gpm                  S90crond
S90xfs                  S95anacron              S95atd
S96readahead            S97messagebus           S97rhnsd
S98cups-config-daemon   S98haldaemon            S99local
```

รายชื่อไฟล์ทั้งหมดที่เห็นใน /etc/rc.d/rc5.d นั้นเป็นเพียง link เท่านั้น คือจะคล้ายๆ กับ short-cut ที่หลายๆ คนคุ้นเคยมากับ MS-Windows อย่างนั้นเอง โดยที่ไฟล์ตัวจริงๆ ของทั้งหมดจะถูกเก็บไว้ใน /etc/rc.d/init.d ซึ่งเป็น script ที่ใช้งานร่วมกันสำหรับทุกๆ run level ... และใน directory ของแต่ละ run level เราก็จะเห็น links ที่คล้ายๆ กันอย่างนี้ทุก directory ... แตกต่างกันก็เพียงแต่ตัว K และตัว S กับตัวเลขกำกับลำดับตรงต้นชื่อของไฟล์เท่านั้นที่อาจจะมากน้อยไม่เท่ากัน

ไอ้ที่เล่ามาซะยืดยาวก็เพราะอยากจะให้เห็นไส้ในตรงนี้แหละครับ ผมชอบแนวคิดที่เขานำมาใช้ในระบบสั่งการของ Linux แบบนี้มากทีเดียว ... คือเขาทำการแบ่งงานออกเป็นส่วนๆ ซึ่งเขียนสั่งเอาไว้

ด้วย script ย่อยๆ หลายๆ ตัวแทนที่จะเขียนเป็นชุดคำสั่งต่อเนื่องที่ยาวๆ จากนั้นก็จับมาเรียงร้อยเข้าด้วยกันผ่าน "*เลขลำดับ*" ที่กำกับไว้หน้าชื่อ เพื่อบังคับให้ชุดคำสั่งแต่ละตัวทำงานตามลำดับก่อนหลังอย่างที่ออกแบบเอาไว้ ... *การจะสลับลำดับการทำงานของแต่ละท่อนจึงง่ายมากหาก user อยากจะทดลอง* :-) ... สำหรับตัวอักษรกำกับตรงหน้าชื่อคือตัว K นั้นย่อมาจากคำว่า Kill หมายถึงชุดคำสั่งที่ต้องการให้ Linux กำจัด process บางอย่างออกไปก่อนที่จะมีการเปลี่ยนแปลงไปสู่ run level หนึ่งๆ (มันคือเหตุผลที่ทำให้มีจำนวนไฟล์ไม่เหมือนกันใน directory ของแต่ละ run level ไง!!) และตัว S ย่อมาจากคำว่า Start ก็คือชุดคำสั่งที่สั่งให้ Linux เรียกใช้ module ต่างๆ ของมันที่จำเป็นสำหรับการใช้งานของแต่ละ run level ... การ Kill หรือการ Start ก็จะเรียงตามลำดับก่อนหลังในหมวดหมู่ของมันตาม "*เลขลำดับ*" ของชื่อไฟล์อย่างเป็นระเบียบเรียบร้อยอย่างที่สุด ... การเขียน script ตัวใหม่ๆ เพิ่มเติมขึ้นมาโดยไม่ต้องแตะต้อง script มาตรฐานเดิมของระบบ จึงเป็นเรื่องที่สามารถทำได้อย่างอิสระ และสามารถกำหนดให้ script ตัวใหม่ของ user ถูกเรียกใช้งานก่อนหรือหลังชุดคำสั่งมาตรฐานตัวใดก็ได้อีกเช่นกัน โดย user ก็ทำเพียงแค่กำหนดชื่อไฟล์ให้ถูกหมวดหมู่ของ Kill หรือ Start พร้อมทั้งกำหนด "*เลขลำดับ*" ที่ต้องการลงไปให้เหมาะสม ... เท่านั้นเอง :-)

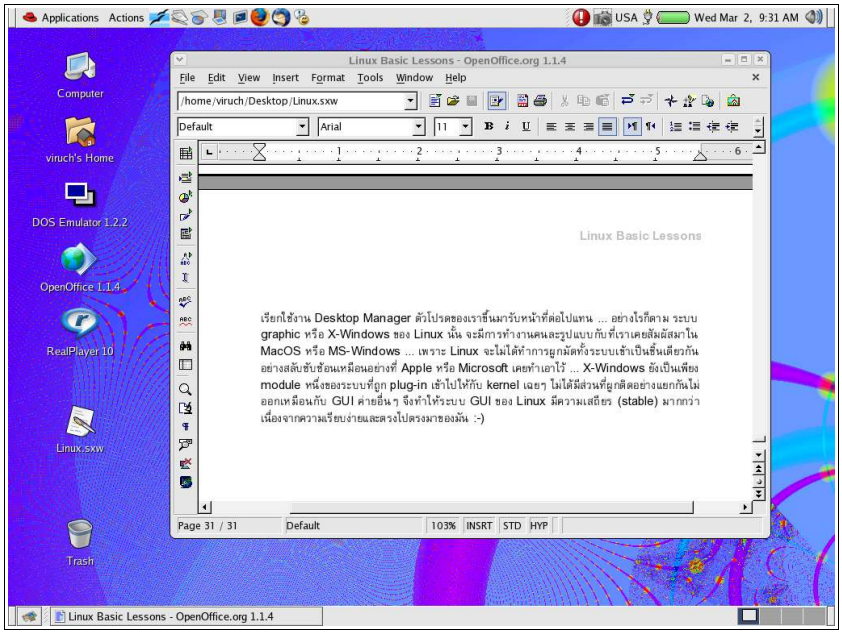
เราจำเป็นที่จะต้องระลึกไว้เสมอว่า Linux นั้นเติบโตขึ้นมากับมือไม้ของเหล่า Hackers และบรรดานักพัฒนา software ทั้งหลายทั่วโลก แม้ว่ามันจะมีชุดคำสั่งที่เป็นภาษา binary ติดมากับระบบอย่างค่อนข้างหลากหลายแล้วก็ตาม แต่ทั้งหมดนั้นต้องถือว่าเป็นเพียง building blocks หรือตัว jigsaw ให้กับพวกเรามาหาวิธีเล่นกันต่อด้วยตัวเอง ... ดังนั้นเอง อาวุธสำคัญที่สุดของ Linux ก็คือ text editor ตัวเก่งที่ใช้กันอย่างแพร่หลายตั้งแต่สมัยเริ่มต้นของ UNIX และยังเป็น text editor ที่ใช้งานกันอยู่จนกระทั่งปัจจุบันในเครื่องระดับ hi-end ทั้งหมดด้วย ... มันคือ vi ซึ่งถูกพัฒนาต่อมาเป็น vim แล้ว ... แต่ทุกๆ คนก็ยังรู้จักมันในชื่อเดิมเสมอ

Linux ในมือของเหล่าโปรแกรมเมอร์นั้นไม่ได้ต้องการอะไรที่มากไปกว่า text editor เก่งๆ ซักตัวหนึ่งเพื่อเอาไว้เขียนโปรแกรม หรือแม้แต่นักเขียนหนังสือบางคนก็ไม่ได้ต้องการอะไรที่หรูหราไปกว่านั้น เนื่องจากการส่งต้นฉบับให้กับโรงพิมพ์ด้วยไฟล์ที่เถื่อนดิบที่สุดจะช่วยลดความสลับซับซ้อนของการโอนชิ้นงานข้ามไปข้ามมาระหว่าง software ที่แตกต่างกันได้อย่างมหาศาล ... และที่สำคัญก็คือโลกของ Linux หรือเครื่องคอมพิวเตอร์ระดับ hi-end ทั้งหมด จะ process งานของมันผ่านไฟล์ข้อมูลที่เรียบๆ ง่ายๆ ในแบบของ text file เท่านั้น เพื่อที่มันจะสามารถส่งผ่านข้อมูลหนึ่งๆ ไปยังเครื่องลูกข่ายที่อาจจะต่างเผ่าต่างพันธุ์กันในโลกของ user ทั่วๆ ไปได้อย่างไม่มีอุปสรรคทางโครงสร้าง

อย่างไรก็ตาม เมื่อ Linux มีอายุอานามที่แก่กล้าเจริญวัย และมีคนเรียกร้องต้องการที่จะทำงานกับมันในรูปแบบที่ง่ายกว่านักพัฒนา software ทั่วๆ ไป ... จึงได้เกิดการพัฒนาระบบ graphic ที่สวยสดงดงามให้กับมันเพิ่มเติมขึ้นมาจากรากฐานดั้งเดิมแบบ pure text ของ UNIX และ Linux ... เขาเรียกระบบ graphic นั้นว่า X-Windows ... ในทำนองว่าเพื่อเป็นการให้เกียรติแก่ต้นตำรับ Graphic User Interfaced (GUI) รายแรกของโลก ... นั่นก็คือ Xerox ... เจ้ายุทธจักรตัวจริงที่ถูก Apple และ Microsoft ช่วยกันขโมยผลงานบันลือโลกชิ้นนั้นมาอย่างหน้าตาเฉย!! ... X-Windows กลายมาเป็นระบบที่สนับสนุนการแสดงภาพ graphic ให้กับ Desktop Manager อันหลากหลายของ Linux แม้ว่าในที่สุดแล้วจะมีที่ได้รับความนิยมแพร่หลายกันอยู่เพียง 2 ตัวคือ KDE และ GNOME ก็ตาม

การ boot ระบบของ Linux นั้นจะไล่ทำงานไปตาม script ที่กำหนดเอาไว้ แล้วใน run level 5 ที่เป็นค่ามาตรฐานของเครื่องส่วนใหญ่ในปัจจุบันก็จะจบลงที่การ load ตัวเองเข้าสู่ X-Windows แล้ว

เรียกใช้งาน Desktop Manager ตัวโปรดของเราขึ้นมารับหน้าที่ต่อไปแทน ... อย่างไรก็ตาม ระบบ graphic หรือ X-Windows ของ Linux นั้น จะมีการทำงานคนละรูปแบบกับที่เราเคยสัมผัสมาใน MacOS หรือ MS-Windows ... เพราะ Linux จะไม่ได้ทำการผูกมัดทั้งระบบเข้าเป็นชิ้นเดียวกันอย่างสลับซับซ้อนเหมือนอย่างที่ Apple หรือ Microsoft เคยทำเอาไว้ ... X-Windows ยังเป็นเพียง module หนึ่งของระบบที่ถูก plug-in เข้าไปให้กับ kernel เฉยๆ ไม่ได้มีส่วนที่ผูกติดอย่างแยกกันไม่ออกเหมือนกับ GUI ค่ายอื่นๆ จึงทำให้ระบบ GUI ของ Linux มีความเสถียร (stable) มากกว่า เนื่องจากความเรียบง่ายและตรงไปตรงมาของมัน :-)



แล้วนี่ก็คือการ dump หน้าจอที่กำลังใช้งานอยู่ โดยจะเห็นไฟล์เอกสารฉบับนี้โชว์หราไว้ด้วย เพราะนี่คือเอกสารฉบับแรกที่สร้างขึ้นมาภายใต้ระบบปฏิบัติการ Linux ล้วนๆ

เราคงจะไม่ลังลึกลงไปกับเรื่องของ X-Windows มากกว่านี้ เพราะระบบ GUI นั้นถูกออกแบบมาจากแนวคิดที่จะให้ user สามารถสื่อสารกับคอมพิวเตอร์ได้ง่ายๆ อยู่แล้ว นอกจากนั้น GUI ของ Linux ก็เป็นสิ่งที่เกิดขึ้นมาเพื่อ "*สนองตลาด*" ไม่ใช่เกิดขึ้นมาจากความตั้งใจเดิมของบรรดานักพัฒนาระบบในยุคยุคแรกๆ ... features หลายๆ อย่างของ GUI ในโลกของ Linux จึงมีความสามารถจำกัดจำเขี่ยอยู่แค่ระดับการใช้งานพื้นฐานของสำนักงานทั่วๆ ไปเท่านั้นเอง และยังต้องรอการพัฒนาที่หลากหลายกว่าที่เป็นอยู่นี้ซะก่อน อิทธิฤทธิ์ด้าน GUI ของ Linux จึงจะเก่งอย่างที่หลายๆ คนคาดหวัง ... ดังนั้นเอง แม้ว่า GUI ของ Linux จะมีความสวยงามและ "*สนองตลาด*" ได้ที่ระดับหนึ่ง แต่ความสามารถที่แท้จริงของ Linux ยังอยู่ในรูปของ command line มากกว่าการสั่งงานด้วย mouse อย่างที่นิยมกัน เพราะฉะนั้น มันจึงเป็นเรื่องที่หลีกเลี่ยงไม่ได้ที่เราจำเป็นต้องเรียนรู้คำสั่งพื้นฐานแบบ command line ของ Linux เอาไว้ เพื่อจะสามารถจัดการกับเครื่องคอมพิวเตอร์ของเราได้อย่างสะดวกและรวดเร็วมากขึ้น

แต่ก่อนที่จะกระโดดไปเล่าเรื่องของคำสั่งพื้นฐานในระบบของ Linux ผมคิดว่าเราน่าจะทบทวนเรื่องที่เล่ามาทั้งหมดอีกทีดีกว่ามั้ย ... คือว่าจริงๆ แล้วเราเพิ่งจะจบไปแค่ 2 หัวข้อเองนะ คือเรื่องของ Partitions แล้วก็เรื่องของ Bootup Sequence :-) ... เรียกว่าเพิ่งจะผ่านเรื่องของการติดตั้งแล้วก็

boot ระบบขึ้นมาเท่านั้นเองนะ การที่ผมจะต่อไปที่เรื่องของคำสั่งพื้นฐานมันก็สมควรอยู่แล้ว ... ใช่รึเปล่า?!

ในเรื่องของ Partitions นั้น Linux บังคับว่าอย่างน้อยต้องมี 1 partition ที่เรียกว่า root หรือแทนด้วยเครื่องหมาย / ... แต่โดยมากเราจะจัดการแบ่งออกเป็น /, /boot, /usr, /var, /tmp และ /home หรือบางครั้งก็จะใมีคนแบ่งออกเป็น /usr/local เพิ่มไว้อีกก็ได้ ทั้งนี้เพื่อความเป็นระเบียบเรียบร้อยและความปลอดภัยของระบบ

ภายใน / หรือ root partition นั้นจะต้องประกอบไปด้วย 5 directories ย่อยคือ /bin, /sbin, /dev, /lib, และ /etc ... ถือเป็น 5 directories ที่ไม่สามารถจัดวางไว้คนละ partition อย่างเด็ดขาด

/boot ทำหน้าที่แค่เก็บรักษา boot images ของ Linux kernel ไว้เท่านั้น

/usr เป็นพื้นที่สำหรับติดตั้งโปรแกรมประยุกต์หรือ applications ต่างๆ ซึ่งโดยมากแล้วในระบบการติดตั้งมาตรฐาน applications ส่วนใหญ่จะติดตั้งลงไปใน /usr/local นั่นคือเหตุผลที่บางคนนิยมแยก /usr/local ออกไปต่างหากอีก 1 partition

/var เป็นพื้นที่ส่วนที่ใช้เพื่อการสื่อสารไม่ว่าจะเป็น e-mail, system log files, และ web server รวมไปถึง cache ที่เกิดจากการ download ไฟล์ต่างๆ ด้วย

/tmp เป็นพื้นที่สำหรับเก็บขยะประเภท temporary files ทั้งหลายของระบบ

/home คือพื้นที่ใช้งานเฉพาะตัวของ user แต่ละคน ส่วนใหญ่ก็จะเอาไว้เก็บไฟล์ข้อมูล และ config ที่ตั้งค่าเอาไว้แบบ customized ของ applications ต่างๆ

เมื่อติดตั้งระบบเรียบร้อยแล้ว Linux ก็จะ boot ตัวเองขึ้นมาใหม่เพื่อให้เรา set ค่าบางอย่างอีกเล็กน้อยก่อนที่จะเข้าสู่ระบบจริงของมัน และในครั้งต่อๆ ไป การ boot ระบบของ Linux ก็จะดำเนินไปตามขั้นตอนปกติของมันคือ

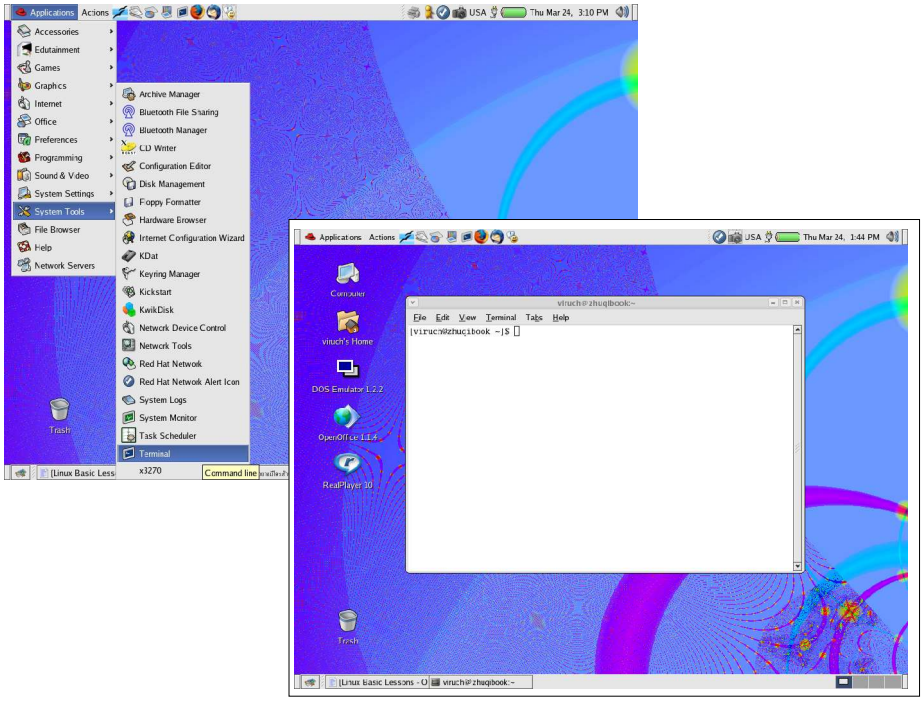
1. BIOS initialization
2. Boot Loader
3. Kernel initialization
4. init starts and enters desired run level by executing:
   - /etc/rc.d/rc.sysinit
   - /etc/rc.d/rc and /etc/rc.d/rc?.d
   - /etc/rc.d/rc.local
   - X Display Manager if appropriate

ในทุกขั้นตอนที่ว่ามานี้ ไฟล์ที่เป็นหัวใจสำคัญในขั้นตอนสุดท้ายของการ boot ก็คือ /etc/inittab, และ /etc/rc.d/rc.sysinit ซึ่งจะไปเรียกใช้งาน script ที่ชื่อว่า /etc/rc.d/rc และ /etc/rc.d/rc?.d ... ทั้งนี้ user ยังสามารถเพิ่มเติม script ของตัวเองไว้ใน /home ของตัวเองได้ด้วย โดย Linux จะเรียก script ที่เขียนเพิ่มเติมนี้จากใน script ที่ชื่อว่า /etc/rc.d/rc.local ... ให้สังเกตด้วยว่าทุกๆ script ที่เอ่ยถึงนั้นจะถูกเก็บอยู่ใน directories ย่อยของ /etc ทั้งสิ้น ... และเมื่อทุกอย่างเสร็จสิ้นสมบูรณ์แล้ว Linux ถึงจะ load ส่วนที่เป็น graphic ของมันที่เรียกว่า X-Windows ขึ้นมา ขึ้นอยู่กับว่าเราเลือกใช้ run level ไหน

พื้นฐานการใช้งานทั่ว ๆ ไปของ Linux

หลังจากที่เราทำการติดตั้ง Linux เรียบร้อยแล้ว เครื่องก็จะถูก boot กลับขึ้นมาใหม่และจะมาหยุดให้เราเติมชื่อของ user และรหัสผ่าน หรือ password เพื่อจะเข้าไปใช้งานระบบ ... โดยปกตินั้น เราจะถูกบังคับให้ต้องบันทึก password ของ root กันตั้งแต่ระหว่างการติดตั้งอยู่แล้ว (ชื่อว่า root นี้เป็น user's name พิเศษที่ไม่สามารถแก้ไขหรือเปลี่ยนแปลงใดๆ ในระบบของ Linux จะถือว่าเป็น super-user หรือ system admin) ... และหลังจากการติดตั้งครั้งแรก เราก็จะมีโอกาสเพิ่มเติมชื่อ user อื่นที่ไม่ใช่ root อีกอย่างน้อย 1 ชื่อ ... เพราะฉะนั้นผมก็จะไม่เริ่มจากการเพิ่มชื่อ user เหมือนกับตำราอื่นๆ เพราะการเพิ่มชื่อให้กับ user นั้นผมถือว่าเป็นธุระของ admin หรือ root ซึ่งจะมีคำสั่งอื่นๆ และส่วนประกอบอื่นอีกหลายตัวที่ต้องพูดถึง ก็เลยจะเก็บรวบยอดไปเล่าทีหลังดีกว่า

ตอนนี้ ผมจะสมมุติว่าพวกเราเข้ามาสู่ตัวระบบเรียบร้อยแล้ว ...



ปกติหลังจากที่ login เข้าไปในหน้า graphic ของ Linux มันก็จะโล้นๆ เกลี้ยงๆ กับมี icons อยู่ซัก 3-4 icons มีแถบเมนูที่อาจจะอยู่ด้านล่างหรือด้านบนก็แล้วแต่ความถนัดของแต่ละคน ... ตรงนี้ไปหาทดลองกันเอาเองแล้วกัน ทดลองจับ mouse แล้ว click ไป click มา เดี๋ยวก็คุ้นไปเองแหละ ... ซึ่งในกรณีของ user ทั่วๆ ไปเขาก็เริ่มใช้งานกันดุ่ยๆ จากตรงนี้เลย ไม่ต้องเรียนรู้อะไรอีก มีเมนูแค่ไหน มี icons อะไรก็ "*ถู่ซี่*" กันไปแค่นั้น (คำว่า "*ถู่ซี่*" คือการออกเสียงที่ถูกต้อง แต่คนไทยชอบทำเป็นมีเสน่ห์จากการออกเสียงไม่ชัดว่า "*ทู่ซี้*" ทั้งๆ ที่ระบบอักขระของเราสามารถเลียนเสียงได้สมบูรณ์อยู่แล้ว)

ส่วนประดาสาวก "*ลัทธิมือคัน*" ให้ไป click ตรง icon ที่มีคำอธิบายว่า Terminal ในแถบ Menu Bar หรือหาก Linux ตัวที่ติดตั้งนั้นไม่ยอมสร้าง icon นี้ไว้ให้ ก็ให้ไปที่เมนู System Tools แล้วเลือกตัวที่มีคำอธิบายว่า Terminal ก็ได้ ... แล้วก็จะมี window ขนาดกลางๆ ปรากฏขึ้นมาอย่างที่เห็นในรูป ... แต่ถ้าชอบแบบ pure text เลยนะ ให้กด Ctrl-Alt-<Function Key> ... โดยสามารถ

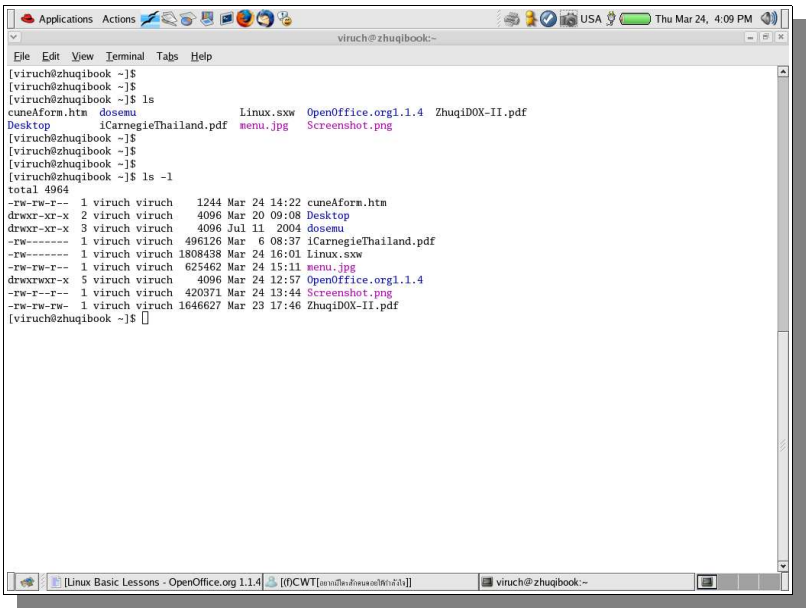
เลือกกดได้ตั้งแต่ F1 จนถึง F6 ... ส่วน F7 จะเป็นหน้าจอที่ระบบจองเอาไว้เป็น graphic mode โดยเฉพาะ (ไม่ได้ dump หน้าจอที่เป็น pure text ไว้ให้นะ เพราะดูเหมือนจะทำไม่ได้)

ที่ต้องพาออกมาดูในส่วนที่เป็น Terminal หรือ Text Mode ของ Linux ก็เพราะตรงส่วนนี้ถึงจะมีอะไรให้เรียนรู้เยอะกว่า แล้วก็ถ้าไม่ลืมกันน่ะนะ ผมก็เคยตั้งข้อสังเกตไว้ว่า Linux นั้นมีพื้นเพดั้งเดิมมาจาก command line เพียง mode เดียว และพัฒนาส่วนที่เป็น graphic ขึ้นมาสนองความต้องการของตลาดเท่านั้น โดยความสามารถที่มีทั้งหมดยังไม่ได้รับการพัฒนาเป็น graphic interfaced อย่างสมบูรณ์เท่าที่ควร ... ดังนั้น graphic mode ของ Linux ก็สนองความต้องการของ user ไปได้กลุ่มหนึ่ง (ประมาณ 70%-80% ของคอมพิวเตอร์ในสำนักงานทั่วๆ ไปเท่านั้น) แต่ก็ยังขาดๆ เกินๆ สำหรับ user อีกกลุ่มหนึ่งที่อยู่ในหน้าที่ของ admin กับการใช้งานแบบเฉพาะกิจเฉพาะทางบางจำพวกหรือพวกบ้า software ประเภทที่ชอบมีให้ครบทุกตระกูลแต่ไม่เคยใช้งานจริงหลากหลายขนาดนั้น (อย่างใน Microsoft Windows ที่เครื่องของผม เป็นต้น)

คำสั่งพื้นฐานจริงๆ ที่ไม่มีทางเลี่ยงเลยก็คือ การขอดูรายชื่อของไฟล์, การ copy ไฟล์, การสร้างไฟล์, การลบไฟล์, แล้วก็คำสั่งที่เกี่ยวกับ directory ต่างๆ ... ซึ่งในโลกของ Microsoft Windows เขาก็พยายามทำเป็นลืมๆ จนไม่ยอมพูดถึงกันอีกแล้ว เพราะมัวแต่ไปใช้งานแบบ graphic ที่อุ้ยอ้ายกว่าในหลายๆ ลักษณะของงาน ... เพราะฉะนั้นผมก็จะไม่ต้องไปเปรียบเทียบกับอดีตอย่าง DOS ว่าเป็นยังไงมายังไงอีกหรือว่าเคยมีการใช้คำสั่งแบบไหน ... เราจะว่ากันเฉพาะ command line ของ Linux เพียง OS เดียวไปเลยดีกว่า

เอาล่ะนะ ... คำสั่งแรกก็คือ ls (แอล-เอส ที่เป็นตัวพิมพ์เล็กนะ)

ls นี่คือคำสั่งเพื่อขอดูรายชื่อของไฟล์ที่อยู่ใน directory ต่างๆ ปกติก็จะเรียกใช้แค่ ls เฉยๆ แต่หากต้องการเห็นรายละเอียดของไฟล์ต่างๆ ด้วยก็จะเติม parameter ต่อท้ายเล็กน้อยคือ -l (ขีด-แอล)



ในรูปที่เห็นนั่นก็คือผลลัพธ์ของวิธีสั่ง ls กับ ls -l ... ครั้งแรกจะเห็นแต่ชื่อไฟล์เรียงไปตาม

แนวขวาง แต่พอสั่งด้วย ls -l มันก็จะเรียงชื่อไฟล์ลงมาทางแนวตั้ง พร้อมๆ กับแสดงรายละเอียดต่างๆ ของไฟล์เช่นว่าใครเป็นเจ้าของ, ขนาดกี่ bites, ถูกสร้างขึ้นมาเมื่อไหร่เอาไว้ด้วย ... เราจะข้ามรายละเอียดพวกนี้ไปก่อนนะ เดี๋ยวจะมีเรื่องเล่าอีกเยอะ

cp คือคำสั่งที่ใช้ในการคัดลอก หรือ copy ไฟล์ รูปแบบในการใช้หรือ syntax ของมันก็ง่ายๆ แค่ว่า **cp <source> <target>** ... อ่านกันง่ายๆ แบบไทยๆ ว่า "*ก็อปปี้จากไหนไปไหน*" เท่านั้นเอง ... เช่นสมมุติว่าจะ copy ไฟล์ชื่อว่า ZhuqiDOX-II.sxw ไปเป็น ZqDX2.sxw เราก็สั่งอย่างนี้ครับ ...

cp ZhuqiDOX-II.sxw ZqDX2.sxw

อย่างนี้เป็นต้น หลังจากกด <Enter> เราก็จะได้ไฟล์ 2 ตัวที่เหมือนกันแต่ใช้ชื่อแตกต่างกัน

rm คือคำสั่งที่ใช้เพื่อการลบไฟล์ ปกติก็จะใช้แค่ **rm <ชื่อไฟล์>** ... ซึ่งการจะใช้งานคำสั่งนี้ต้องระมัดระวังซักหน่อย เพราะว่าระบบปกติของ Linux นั้น การลบไฟล์ใน text mode คือการลบจริง ไม่ใช่แค่เอาใส่ใน trash แบบที่เห็นใน graphic mode ซึ่งเรายังมีโอกาสกู้คืนกลับมาได้บ้าง (ถ้าจะให้ text mode มี feature คล้ายๆ อย่างนี้ต้องมีการกำหนดค่าบางอย่างไว้ต่างหาก

ในกรณีของการสร้างไฟล์นั้น หากไม่มีการใช้ software อย่างอื่นเข้ามาปะปนเลย มันก็จะเป็นการสร้าง text file ธรรมดาเท่านั้น ซึ่งจริงๆ แล้ว Linux ก็จะมีคำสั่งในการสร้างไฟล์เล็กๆ ประเภท line editor อยู่บ้างเหมือนกัน แต่ผมจำชื่อไม่ได้แล้วล่ะ เพราะส่วนใหญ่ของการสร้าง text file ในโลกของ Linux นั้น เขาจะใช้อภิมหาอมตะนิรันดร์กาลอย่าง vi เพียงหนึ่งเดียวเท่านั้นเอง

การสร้างแค่ text file อาจจะฟังดูเป็นเรื่องบ้าๆ บอๆ สำหรับคนที่เติบโตมากับ software เถื่อนที่สามารถสร้างงานได้สารพัดชนิดด้วยเครื่องคอมพิวเตอร์ประจำตัว ... แต่สำหรับ Linux ที่เป็นอีกโลกของเหล่าโปรแกรมเมอร์หรือ developers แล้วนั้น แค่ text file ธรรมดาๆ ที่ผูกโยงกันไว้ด้วยไวยากรณ์ทางโปรแกรมคอมพิวเตอร์ภาษาใดภาษาหนึ่ง ก็อาจจะก่อให้เกิดผลลัพธ์ที่ user ธรรมดาๆ อย่างเราๆ ไม่มีทางเข้าใจได้เลย

สำหรับ vi ที่เป็น text editor ยอดนิยมของชาว Linuxian นั้น ผมคิดว่าจะไม่เล่าไว้เลยดีกว่า เพราะรายละเอียดเยอะขนาดที่สามารถเขียนเป็นตำราต่างหากได้เล่มหนึ่งเลยทีเดียว ประกอบในระบบปกติที่ติดตั้ง Linux ไว้แล้วนั้น จะมีคำสั่ง vimtutor เอาไว้เรียนรู้และฝึกหัด vi อยู่แล้วด้วย ... แล้วก็ที่สำคัญสุดๆ เลยนะ ... vimtutor ที่ว่านั้นเป็นแค่ text file ธรรมดาๆ ที่ผูกโยงกันไว้ด้วยไวยากรณ์พิเศษชนิดหนึ่งที่เรียกว่า shell script ... หนึ่งในอานุภาพที่ร้ายกาจที่สุดของ Linux เลยทีเดียว

mv เล่าเรื่องคำสั่งพื้นฐานต่อ ... mv เป็นคำสั่งเพื่อใช้ในการ "*ย้ายไฟล์*" ให้ไปเก็บไว้ใน directory อื่น หรืออาจจะเป็น "*ชื่ออื่น*" ก็ได้ ... อย่าลืมว่า Linux จัดการทุกเรื่องอยู่บนพื้นฐานของ filesystems เท่านั้น คือมองทุกๆ องค์ประกอบเป็น "*ไฟล์*" ไปหมด ... รูปแบบของคำสั่ง mv ที่เป็น **mv <source> < target>** นั้น หากเรากำหนด <target> ให้เป็นเพียงชื่อไฟล์อีกชื่อหนึ่ง Linux ก็จะถือว่าเป็นการ "*เปลี่ยนชื่อ*" แล้วก็จัดเก็บไฟล์นั้นๆ ไว้ที่ directory เดิม ... แต่หาก <target> มีการกำหนดให้มีชื่อของ directory กำกับไว้พร้อมกับชื่อไฟล์ Linux ก็จะถือว่าเป็นการย้ายที่เก็บไฟล์ ส่วนว่าจะเป็นชื่อเดิมหรือไม่ก็ขึ้นอยู่กับการกำหนดของเราอีกเช่นกัน

คำสั่งพื้นๆ ที่เกี่ยวกับไฟล์ของ Linux ก็จะมีกันแค่ 4 คำสั่งนี้เป็นหลักเท่านั้นแหละครับ ทีนี้เราก็มาถึงเรื่องของ directory กันมั่ง

mkdir ใช้ในการสร้าง directory ใหม่ วิธีใช้ก็แค่สั่ง **mkdir <ชื่อ directory>**

rmdir ใช้ในการลบ directory ใด directory หนึ่ง รูปแบบคำสั่งก็จะคล้ายๆ กับตอนที่สร้างคือ **rmdir <ชื่อ directory>** ... แต่หาก directory นั้นๆ มีไฟล์หรือ sub-directory อยู่ด้วย เราก็จะต้องไล่ลบไฟล์หรือ sub-directory ย่อยๆ ทั้งหมดออกไปก่อน จึงจะสามารถลบ directory ระดับบนสุดออกไปได้ ... ซึ่งเป็นวิธีการที่โบร่ำโบราณแต่ก็ปลอดภัยดี แต่สำหรับคนที่ชอบทำอะไรๆ ให้รวบรัด เราจะใช้คำสั่งว่า **rm -rf <ชื่อ directory>** แทน ซึ่งจะลบทั้งไฟล์ใน directory นั้นๆ รวมทั้งตัว directory ที่ระบุออกไปทั้งหมดพร้อมๆ กัน ... แล้วก็อย่าลืมว่ามันกู้คืนไม่ได้นะ โปรดใช้อย่างระมัดระวังสุดๆ โดยเฉพาะ rm -rf * จะรุนแรงขนาดทำให้ทุกอย่างพังพินาศได้ในเวลาเพียงพริบตาเท่านั้นเอง

cd ใช้ในการย้ายตำแหน่งของการปฏิบัติงานของเราไปยัง directory ต่างๆ ของระบบ คำสั่งก็ไม่มีอะไรมาก ก็แค่ **cd <ชื่อ directory>** เท่านั้นครับ

ถือว่าทบทวนความจำกันตรงนี้นิดหนึ่ง สำหรับเครื่องหมายที่ใช้ขั้นระหว่างชื่อไฟล์กับชื่อ directory ในระบบของ Linux ... เขาใช้เครื่องหมาย slash (/) เป็นตัวขั้นนะครับ ตัวอย่างเช่น

/home/viruch/dosemu หรือ
/usr/share/wallpapers/ZhuqiFract.bmp อย่างนี้เป็นต้น

ดังนั้น เวลาจะ cd ไปไหนต่อไหน หรือจะระบุชื่อไฟล์อะไรตรงไหนของระบบ ก็ต้อง key ให้มันครบๆ และสะกดอย่างถูกต้องพร้อมกับต้องใช้ตัวพิมพ์ใหญ่-พิมพ์เล็กให้แม่นยำเสมอด้วย ... ตรงนี้ของแทรกชื่อ directory พิเศษไว้ 3 ตัวด้วยครับ

. (จุดเดียว) จะหมายถึง current directory ที่เราทำงานอยู่

.. (2 จุด) จะหมายถึง parent directory ของ directory ปัจจุบันของเรา

~ (เครื่องหมาย tile) หมายถึง home directory ของ user นั้นๆ ที่ใช้งานระบบอยู่

ตัวสุดท้ายที่ถือว่าพื้นๆ และจำเป็นพอสมควรก็คือ **exit** ที่เราเอาไว้ออกจาก Terminal หรือคำสั่ง **logout** ในกรณีที่เรากระโดดออกไปที่ text mode จริงๆ ด้วยการกด Ctrl-Alt-<Fn-Key> ตั้งแต่แรกเลย ... เพราะผมนึกอยากจะหยุดตรงนี้ก่อนแล้วล่ะ :-)

คำสั่งพื้นฐานมากๆ กลุ่มนี้ ก็คือสิ่งที่ user ทุกคนต้องรู้ก่อนที่จะเริ่มออกทัวร์ในระบบ filesystems ของ Linux ในตอนต่อๆ ไปครับ คนที่คุ้นเคยกับคอมพิวเตอร์มานานโดยเฉพาะกับ MS-DOS ที่มีคำสั่งคล้ายๆ จนเกือบจะเหมือนกันเลยนั้น อาจจะรู้สึกว่าคำสั่งเหล่านี้ไม่คู่ควรกับการเปลืองกระดาษ ... แต่สำหรับคนที่แทบจะคลั่งใจตายเวลาเห็น prompt กระพริบๆ อยู่ตรงหน้าจอนั้น ... ช่วยกันจำไว้หน่อยก็จะดีนะครับ เพราะผมคิดว่าถึงเวลาออกทัวร์กันจริงๆ แล้วล่ะ :-)

ประถมบทของ ***Linux***

อยากจะบอกว่าที่เล่าไปเมื่อไม่กี่คำสั่งที่ผ่านมานั้น ผมถือว่าเป็นความรู้ในระดับ "*อนุบาล*" เท่านั้นเอง แบบเดียวกับที่เด็กๆ ต้องรู้วิธีช่วยเหลือตัวเองขั้นพื้นฐานไว้บ้างก่อนที่จะเริ่มก้าวเข้าสู่ระบบการศึกษาจริงๆ ... หลังจากนั้น เด็กๆ ก็จะเริ่มงงกับสิ่งที่ผ่านเข้ามาในชีวิตของตัวเองง่ายขึ้นเรื่อยๆ :-)

"*การงง*" เป็นอาการที่ดีและมีประโยชน์ต่อระบบการเรียนรู้ของมนุษย์เป็นอย่างมาก เพราะขณะที่เราเริ่มงงนั้น จะหมายถึงกลไกในสมองเริ่มที่จะตอบสนองต่อสิ่งเร้าภายนอก และมีการประมวลผลเพื่อแยกแยะระหว่างสิ่งที่รู้แล้วกับสิ่งที่ยังไม่รู้ สมองจะพยายามหาจุดต่อรอยเชื่อมเพื่อสังเคราะห์ประสบการณ์ต่างๆ เหล่านั้นเข้าด้วยกัน อันจะนำไปสู่การเกิดขึ้นขององค์ความรู้ใหม่ๆ ... **ยิ่งเด็กๆ เกิดอาการงงได้ง่ายเท่าไหร่ ก็จะยิ่งมีโอกาสเรียนรู้ได้มากขึ้นเท่านั้น** ... ถ้าบังเอิญมันไม่ท้อแท้สิ้นหวังไปซะก่อนนะ :-)

เอาล่ะนะ ... เพื่อที่จะให้พวกเรางงกันได้อย่างอิสระ ผมขอแนะนำให้ใช้คำสั่ง **man <ชื่อคำสั่ง>**

คำสั่ง man ย่อมาจากคำว่า **man**ual มีเอาไว้เพื่อเปิดอ่านคู่มือ หรือเพื่อเรียนรู้วิธีใช้คำสั่งต่างๆ เกือบจะทั้งหมดของ Linux เลยก็ว่าได้ ก็ไปทดลองเปิดคู่มือของ*คำสั่งระดับอนุบาล*ที่เล่าไปแล้วนั่นดูแล้วกัน เพราะมันมี options หรือ parameters ให้เลือกใช้ได้อีกเยอะมาก ทั้งยังสามารถเอา options เหล่านั้นมาผสมผเสกันได้อีกต่างหาก ... เวลาจะออกจาก man ก็ให้กด **q** เพื่อกลับออกมาที่ system prompt ตามเดิม ... ไปลองกันเองนะ ...

ทีนี้ ... ผมจะย้อนกลับไปที่คำสั่ง **`ls -l`**

เวลาที่เราใช้คำสั่งนี้ Linux ก็จะตอบสนองด้วยการแสดงรายชื่อของไฟล์พร้อมรายละเอียดประกอบออกมาเต็มไปหมด โดยในแต่ละบรรทัดที่เห็น ก็จะเป็นรายละเอียดของไฟล์เพียง 1 ไฟล์เท่านั้น ... อะไรถึงจะต้องมากมายกันขนาดนั้น?

-	rw-r--r--	1	root	root	325	Oct 16 07:21	auto.master
d	rwxr-xr-x	2	root	root	4096	Jan 24 04:57	autopackage
l	rwxrwxrwx	1	root	root	22	Jan 18 04:43	grub.config
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**

ยกตัวอย่างมาให้ดู 3 ไฟล์แล้วกัน ผมระบาย shade กับระบุตัวเลขกำกับเป็นแถบๆ เอาไว้นะ จะได้เล่าได้ถนัดๆ หน่อย

column 1 เป็นตัวแสดงคุณสมบัติของไฟล์นั้นๆ ว่าเป็น file (-) , **d**irectory (d) , **l**ink (l) , หรืออาจจะเป็น **b**lock device (b) ... column 1 นี้เป็นอักษรตัวเดียวและมักจะเขียนติดกันพรืดกับ column 2 ไปเลย ไม่ได้เว้นห่างๆ อย่างในตัวอย่างนี้

column 2 เป็นส่วนที่แสดง permission ของไฟล์นั้นๆ ... ตรงส่วนนี้ประกอบด้วยอักษร 3 ชุดที่เหมือนๆ กัน โดยแต่ละชุดก็จะมีอักษรได้ 3 ตัวคือ r w และ x ... คือ rwxrwxrwx ... แต่อาจจะมีการเปิดๆ ปิดๆ สิทธิ์เหล่านี้สลับไปสลับมาดังที่เห็นในตัวอย่างข้างบนนั่น

ชุดแรกเป็นส่วนของ user หรือเจ้าของไฟล์, ชุดถัดมาตรงกลางเป็นของ user's group หรือผู้ใช้อื่นๆ ที่อยู่ใน group เดียวกับเจ้าของไฟล์, ชุดหลังสุดด้านขวาก็จะเป็นเรื่องของ other user หรือผู้ใช้อื่นๆ ที่ไม่ใช่เจ้าของไฟล์ ทั้งยังไม่ได้อยู่ใน group เดียวกับเจ้าของไฟล์อีกต่างหาก

r ที่เห็นนั้นย่อมาจาก **read** หมายถึงสิทธิ์ในการ "*อ่าน*" ไฟล์นั้นๆ หรือ "*เปิด*" ไฟล์นั้นๆ ออกมาดูเนื้อหาข้างใน หรืออาจจะสั่งให้อ่านไปออกที่เครื่องพิมพ์ อย่างนี้เป็นต้น

w ย่อมาจาก **write** หมายถึงสิทธิ์ในการ "*เขียน*" ไม่ว่าจะเป็นการเขียนเพิ่มเติมหรือลบบางสิ่งบางอย่างออก รวมทั้งการสั่งลบทิ้งไปทั้งไฟล์

x ย่อมาจาก e**x**ecute หมายถึงการสั่งให้ไฟล์นั้นๆ ปฏิบัติงานตามโปรแกรมที่ถูกกำหนดเอาไว้ภายในตัวของมัน ... ตรงนี้จะแตกต่างจากโลกของ Microsoft Windows หรือ MS-DOS ที่ความสามารถในการ execute ไฟล์ใดไฟล์หนึ่งนั้น มักจะผูกติดอยู่กับการกำหนดชื่อให้กับไฟล์เสมอเช่นต้องเป็น .com .exe หรือตัวอื่นๆ ที่กำหนดไว้อย่างตายตัวด้วย OS ... แต่ในโลกของ Linux จะกำหนดผ่านระบบ permission ของไฟล์แทน จึงหมายความว่าไฟล์ใดๆ ก็ตามแม้ว่าจะเป็นโปรแกรมที่เขียนเอาไว้อย่างถูกต้อง แต่หากไม่ได้รับการกำหนดสิทธิ์ให้สามารถ execute ไฟล์นั้นๆ เราก็จะไม่สามารถ load ขึ้นมาสู่การปฏิบัติงานใดๆ ได้เลย

มาดูไฟล์ในตัวอย่างกันหน่อย ไฟล์แรกที่ชื่อว่า auto.master นั้นมีข้อมูลที่กำหนดไว้เป็น -rw--r--r-- หมายความว่ามันเป็น "*ไฟล์ธรรมดา*" ที่สามารถอ่านและเขียนได้โดยเจ้าของไฟล์ แต่จะอ่านได้เพียงอย่างเดียวสำหรับคนอื่นๆ ไม่ว่าจะอยู่ group เดียวกับเจ้าของไฟล์หรือไม่ก็ตาม ... ข้ามไปตัวที่ 3 ก่อน มันมีข้อมูลที่กำหนดไว้เป็น lrwxrwxrwx หมายความว่ามันเป็นเพียง link ไม่ใช่เป็นตัวไฟล์จริงๆ และกำหนดให้ทุกๆ user ของระบบสามารถ "*อ่าน*" , "*เขียน*" , หรือ execute ให้มันปฏิบัติงานได้ เพราะเปิดสิทธิ์เอาไว้เต็มทั้ง 3 ชุด

ไฟล์ตัวอย่างที่ 2 นั้นเป็น directory เพราะมี bit แรกของชุดข้อมูลเป็นตัว d โดยมี permission เป็น drwxr-xr-x ซึ่ง permission ของ directory จะเพี้ยนไปจากของไฟล์บ้างเล็กน้อย เพราะ **r** จะหมายถึงมีสิทธิ์มองเห็นชื่อของ directory นี้ และ **w** จะหมายถึงมีสิทธิ์ในการเพิ่มไฟล์หรือลบไฟล์ออกจาก directory นี้ ในขณะที่ **x** จะหมายถึง access หรือสิทธิ์ในการเข้าไปเห็นสิ่งที่ถูกจัดเก็บเอาไว้ภายใน directory นี้เท่านั้น ... แต่ทั้งหมดนั้นก็ยังคาบเกี่ยวกับ permission แบบไฟล์ด้วยเหมือนกัน เช่น directory ที่มี permission เป็น dr-xr-xr-x จะกลายเป็น directory ที่ไม่มีใครสามารถเอามันออกไปจากระบบ เพราะการลบไฟล์หนึ่งๆ จะต้องมีสิทธิ์ w อยู่ใน permission นั้นเสมอ หรือ directory ที่มี permission เป็น d--x--x--x ก็จะกลายเป็น directory ที่ user สามารถเข้าไปได้ แต่จะไม่เห็นไฟล์หรืออะไรอื่นใดข้างในเลย เพราะไม่มีสิทธิ์ในการ read ที่จะต้องกำหนดเอาไว้ตั้งแต่ระดับ directory ด้วยเสมอ ... หรือแม้แต่สิทธิ์ w ที่สามารถเขียนหรือลบไฟล์ก็เหมือนกัน หากไม่ได้รับอนุญาตตั้งแต่ระดับ directory แล้ว user หนึ่งๆ ก็จะไม่สามารถลบไฟล์ใดๆ ออกจาก directory นั้นๆ เลย ไม่ว่า user นั้นจะมีสิทธิ์ในระดับไฟล์มากแค่ไหนก็ตาม :-) ... มึนดีมั้ยล่ะ?

column 3 เป็นจำนวนนับของ hard link ที่ชื่อของไฟล์นั้นๆ มีอยู่ในระบบ (เรื่อง link จะเอาไว้

เล่าทีหลังครับ)

column 4 ชื่อเจ้าของไฟล์ ... การเป็นเจ้าของไฟล์จะหมายถึงสิทธิ์ในการกำหนด permission ให้กับไฟล์นั้นๆ ด้วยคำสั่ง chmod ได้

column 5 ชื่อ group ของเจ้าของไฟล์ หรือ group ที่ต้องการให้เกี่ยวข้องกับไฟล์นั้น

column 6 ขนาดของไฟล์ที่คำนวณเป็น bites

column 7 วันที่และเวลาที่ไฟล์นั้นๆ ถูกสร้างขึ้น หรือหมายถึงวันที่สุดท้ายที่ไฟล์นั้นถูกแก้ไข

column 8 ชื่อของไฟล์ ซึ่งส่วนใหญ่จะใช้ตัวพิมพ์เล็ก เพื่อความสะดวกในการเรียกใช้เรียกหา เนื่องจากระบบการเรียกชื่อไฟล์ของ Linux นั้นจะเป็นพวก case sensitive คือ ไฟล์ที่ชื่อว่า ZHUQ, zhuq, Zhuq, ZHuq, ฯ จะหมายถึงคนละไฟล์ไปเลยในโลกของ Linux :-) เพราะฉะนั้นพึงระมัดระวังในการตั้งชื่อไฟล์ เพราะอาการแบบนี้อาจจะทำให้ Microsoft Windows กลายเป็นบ้าไปได้ง่ายๆ หากจะมีการ share ไฟล์ให้ใช้งานร่วมกัน

ไหนๆ ก็ลากมาจนถึงเรื่อง permission bits กันแล้ว เพราะฉะนั้นผมก็จะเล่าต่อที่เรื่องนี้ไปเลยก่อนที่จะพูดถึงเรื่องอื่นๆ ต่อไป คำสั่งที่เกี่ยวกับ permission bits ก็คือ chmod

chmod เป็นคำสั่งที่มี options ได้หลายรูปแบบ แต่แบบที่นิยมใช้มากที่สุดก็คือแบบที่ใช้ตัวเลขในการกำหนด permission เพราะ key กันน้อยๆ และรวดเร็ว แต่ก็อาจจะเข้าใจยากกว่าแบบที่กำหนดด้วยตัวอักษรตรงๆ ... ทั้งนี้ก็ขึ้นอยู่กับแต่ละสถานการณ์ด้วย เพราะมันจะมีความยากง่ายแตกต่างกันไป

วิธีการกำหนดเป็นตัวเลขคือ r = 4, w = 2, x = 1 ... ก็เลยทำให้ rwx = 7, rw- = 6, r-x = 5

ตัวอย่างเช่น ถ้าจะกำหนด permission ให้เป็น rwxr-xr-x เราก็จะสั่งว่า

chmod 755 <ชื่อไฟล์/ชื่อ directory> แทนที่จะสั่งว่า

chmod u=rwx,g=rx,o=rx <ชื่อไฟล์/ชื่อ directory> หรือ

chmod u+r+w+x,g+r+x,o+r+x <ชื่อไฟล์/ชื่อ directory>

อย่างนี้เป็นต้น :-) ลองดูความรุ่มร่ามของคำสั่งกันเอาเอง ... แต่ทำไม r=4, w=2, x=1 นั้นมันเป็นเรื่องของการแปลงค่าเลขฐานสองครับ ทั้ง r, w, x นั้นหากเปิดสิทธิ์ให้ก็หมายถึง 1 แต่หากปิดไว้ก็หมายถึง 0 ดังนั้น rwx หากเขียนเป็นเลขฐานสองก็คือ 111 เมื่อแปลงกลับมาเป็นเลขฐานสิบจะมีค่าอย่างนี้ครับ หลักแรกมีค่าเท่ากับ $2^0=1$ หลักที่สองมีค่าเท่ากับ $2^1=2$ และหลักที่สามมีค่าเท่ากับ $2^2=4$ อย่างนี้เองแหละครับ เวลากำหนด permission เป็น rwx จึงมีค่าเท่ากับ 4+2+1=7 นั่นเอง

ก็คงต้องนั่งนิ่งๆ แล้วทำความเข้าใจกันหน่อยล่ะครับ หรืออาจจะเปิดดูวิธีใช้คำสั่ง chmod ด้วยการเรียก **man chmod** ดูก็ได้ เนื่องจาก Linux เป็นระบบแบบ multi-users และการกำหนด permission ก็เป็นงานเบื้อต้นของระบบ security ชะด้วย เพื่อที่จะไม่ให้ user ขี้ชั่วไปเขียนหรือไปลบไฟล์ที่เจ้าของงานเขาไม่ยินยอมพร้อมใจ

นอกจาก permission แบบพื้นๆ นี้แล้ว Linux ยังมี bit ที่ซ่อนไว้อีก 1 bit เรียกว่า sticky bit ซึ่งจะสลับซับซ้อนกว่า และผมคิดว่าไม่อยากเล่าด้วยตัวอักษรเพราะมันอธิบายด้วยปากเปล่าจะง่ายกว่า และจริงๆ ก็ยังมีเรื่องของ attribute ของไฟล์ที่ไม่ค่อยจะมีใครพูดถึง เพราะ Linux ที่เติบโตมากับโลก

แบบ community หรือ communist นั้น อาจจะไม่ค่อยพะวักพะวงกับเรื่องของการป้องกันความรั่วไหลหรือเรื่องจุกๆ จิกๆ ใน "*โลกแบบทุนนิยม*" เท่าที่ควร ทั้งๆ ที่มีการกำหนดรายละเอียดบางอย่างเอาไว้แล้วตั้งแต่ในระดับของ kernel ด้วยซ้ำไป ... ผมเองก็เคยเห็นแค่ผ่านๆ และยังอยากจะหาเอกสารเกี่ยวกับเรื่องพวกนี้มาศึกษาเพิ่มเติม เพราะลำพัง permission แค่ 9 bit เท่าที่เห็นนี้ ยังไม่เพียงพอกับความต้องการจริงในโลกของ network หรอกครับ!!

ถัดจากเรื่องของ permission ก็จะมาถึงเรื่องของ user's name กับ user's group ซึ่งตำรา Linux ส่วนใหญ่จะจับมันไปอยู่ท่อนแรกๆ ด้วยซ้ำไป ทั้งๆ ที่มันเป็นงานของ admin หรือ root ไม่ใช่ธุระของ user ทั่วๆ ไป ... ในความรู้สึกของผมเองนะ คำสั่งในการเพิ่มหรือลดหรือการบริหารระบบ user นั้นต้องเอาบทสุดท้ายเลยครับ :-) เพราะหากว่าตัวเราเองยังไม่รู้จะทำอะไรอย่างอื่นต่อ ผมก็ไม่รู้เหมือนกันว่าจะมี user ที่ไหนที่เขาจะยอมมาใช้เครื่องของเรา? แล้วถ้า user นั้นๆ เกิดมีปัญหาขึ้นมาจริงๆ เราเองที่ทะลึ่งติดตั้งระบบบอย่างงูๆ ปลาๆ นี้ ยังจะไปช่วยแก้ปัญหาหรือทำอะไรให้กับเขาได้บ้าง? ... อย่ากระนั้นเลย ... ผมยังไม่อยากเล่าเรื่องของการเพิ่มหรือลดหรือการบริหารระบบ user ในตอนนี้หรอก !!..

ยังไงซะเราก็เป็นคนที่ติดตั้งระบบทั้งหมดด้วยตัวเองเองมาตั้งแต่แรก ดังนั้นก็เป็นเราเองอีกนั่นแหละที่เป็นผู้กำหนด password ของ root หรือ admin ของเครื่องที่เราจัดการอยู่ เราจึงควรจะสนใจรายละเอียดในหน้าที่ของ root ก่อนที่จะไปสนใจว่าใครจะเข้ามาใช้งานในระบบของเราเวลานี้ ... เพราะงั้นผมก็เลยจะย้อนกลับไปให้ไกลกว่า **`ls -l`** อีกซักหน่อย ... ย้อนกลับไปถึงตอนที่ติดตั้งระบบกันเลย ...

จำได้มั้ยว่าผมบอกไว้ยังไงตอนที่เริ่มติดตั้งระบบ ... เราอาจจะกำหนด mount point มันโดดๆ ไปเลยแค่จุดเดียวก็คือ / หรือ root filesystem จากนั้นก็เลือก packages ที่เราต้องการให้หนำใจแล้วก็รอการติดตั้งทั้งหมดประมาณ 1 ชั่วโมง ... นั่นคือการลงทุนเวลา 1 ชั่วโมงที่เราจะได้เรียนรู้อะไรบางอย่างที่สำคัญของ Linux ตัวที่เราเลือก ...

เอาล่ะ !! .. ออกไปที่ Terminal อีกครั้ง แล้วเรียกใช้คำสั่งนี้ครับ

df -hT แล้วกด <Enter> ลงไป เครื่องที่ผมใช้งานอยู่ มันจะแสดงอย่างนี้ ...

Filesystem	Type	Size	Used	Avail	Use%	Mounted on
/dev/hda4	ext3	9.3G	5.3G	3.6G	60%	/
/dev/hda3	ext3	97M	14M	79M	15%	/boot
none	tmpfs	379M	0	379M	0%	/dev/shm

จะเห็นว่า เครื่องที่ผมใช้งานอยู่นั้นมันแบ่ง partitions ไว้แค่ 2 partitions เท่านั้นคือ / กับ /boot ... อันนี้เนื่องจากข้อจำกัดบางอย่างในตอนที่ติดตั้ง เพราะมันเป็นเครื่อง multi-platform ที่ลงคู่กันกับ Microsoft Windows ซึ่งแยก partitions ไว้ค่อนข้างจากมากแล้วก่อนที่จะลง Linux ... เอ้า! .. เราดูกันแค่เห็นๆ นี่แหละ

พื้นที่ของ Root Filesystems หรือ / นั้นผมแบ่งเอาไว้แค่ 9.3GB และถูกใช้ไปแล้ว 5.3GB ...ใช้อะไรไปมั่งเดี๋ยวค่อยว่ากัน ส่วน /boot นั้นตอนที่ติดตั้งผมแบ่งไว้ 100MB แต่มันคำนวณเพี้ยนกันนิดหน่อยก็เลยออกมาแค่ 97MB แต่ก็ใช้งานไปแค่ 14MB เท่านั้นเอง ยังเหลือเนื้อที่อีกบานสำหรับการเก็บแค่ boot images กับ kernel ของ Linux เพราะที่เห็นว่าใช้ไป 14MB นั้นหมายถึงมันมีอยู่

ด้วยกันตั้ง 3 version !! ... ตัวสุดท้ายเป็น tempory directory ของ Linux ที่มันสร้างขึ้นมาเอง จะเห็นว่าประเภทหรือ format ของมันจะเป็น tmpfs หรือ temporary file system

ทีนี้ลองมาดูอีกเครื่องหนึ่งที่เป็น Macintosh แต่เอามาลง YellowDog Linux ซึ่งไม่ได้ติดขัดในข้อจำกัดของการแบ่ง partition แบบ PC ดูมั่ง ... ด้วยคำสั่งเดียวกันผมจะได้รายการอย่างนี้ครับ

Filesystem	Type	Size	Used	Avail	Use%	Mounted on
/dev/hda4	ext3	485M	173M	287M	38%	/
/dev/hda3	ext3	97M	16M	76M	18%	/boot
none	tmpfs	314M	0	314M	0%	/dev/shm
/dev/hda8	ext3	985M	75M	860M	9%	/home
/dev/hda5	ext3	485M	8.1M	452M	2%	/tmp
dev/hda10	ext3	7.9G	4.1G	3.5G	55%	/usr
/dev/hda6	ext3	485M	100M	360M	22%	/var
/dev/hda9	ext3	5.0G	33M	4.7G	1%	/zhuqidisk

จะเห็นว่าผมเล่นตามตำราเลย :-) แยกเป็น /, /boot, /home, /tmp, /usr, /var แถมด้วย partition ประหลาดต่างหาก เพื่อเอาไว้ download ไฟล์ขนาดใหญ่ๆ จาก internet ... ลองดูขนาดของแต่ partition กันเอาเองนะ เรียกว่าเผื่อกันเอาไว้เต็มๆ *(แล้วยังเหลือพื้นที่ว่างอีก 12G ที่ยังไม่ได้แบ่งเป็น partition ใดๆ เลยด้วย)*

แต่นั่นก็แค่คำสั่ง **df** หรือ **d**isk-**f**ree ซึ่งสามารถเรียกใช้งานได้ทุกคน เพราะมันไม่ได้ access เข้าไปในพื้นที่ต้องห้ามของระบบ แค่ตรวจสอบดูขนาดของแต่ละ partition เท่านั้นเอง ... คือถ้าตอนติดตั้งระบบนั้น เราเลือกลงแบบมี / หรือ Root Filesystems เพียงตัวเดียว เราก็จะได้หน้าตาออกมาคล้ายกับตัวอย่างแรก ซึ่งก็นั่นแหละที่เราอยากจะไปรู้ต่อว่า เราควรจะแบ่งเป็น /boot, /home, /usr, /var, /tmp อย่างละประมาณเท่าไหร่

ตอนนี้ถ้าใครไม่ได้ login เข้าระบบด้วย root ก็จัดการ **s**witch **u**ser ไปเป็น root ด้วยคำสั่ง

su -

... อย่าลืม **-** ข้างหลังนะครับ เพราะมันมีบางอย่างไม่เหมือนกันระหว่าง **su** กับ **su –** ... แม้ว่าทั้ง 2 คำสั่งจะเป็นการ switch user ไปเป็น root ด้วยกันทั้งคู่ แต่ **su –** เท่านั้นที่จะไปอ่านค่าตัวแปรหรือ parameters ของ root มาจริงๆ ซึ่งในบางกรณีเราก็จำเป็นต้องใช้เหมือนกันเช่นพวก %path หรืออะไรพวกนั้น ... เพราะฉะนั้นแค่ขีดเดียวก็ใส่ๆ เข้าไปซะ ... อย่างเรื่องมาก!!

จากนั้น cd ไปที่ root directory ด้วยคำสั่ง cd /

แล้วก็เรียกคำสั่งนี้ขึ้นมา ... **du -h --max-depth=1** ... แล้วก็กด <Enter>

เครื่องของเราจะนิ่งไปพักหนึ่งเลยครับ เพราะมันจะต้องไปควานหาและคำนวณขนาดของไฟล์ต่างๆ ที่ติดตั้งไว้ในระบบของเราออกมา ... ก็คงไม่ต้องเปรียบเทียบกับเครื่อง Macintosh แล้วนะครับ เพราะจริงๆ ก็ต้องการจำลองเหตุการณ์ว่า เราได้ติดตั้งแบบใช้ partition เดียวอยู่แล้ว ... อีกอย่าง ... เครื่อง PC ที่ใช้ chip ของ intel ก็หา software มาลองเล่นได้มากกว่าด้วย เพราะฉะนั้นเราก็มาดูซิว่า หลังจากการติดตั้งและผ่านการ update หลายๆ อย่างจนเข้าที่อย่างที่ตัวเองอยากได้แล้วนั้น แต่ละส่วนมันจำเป็นต้องใช้พื้นที่ hard disk ซักเท่าไหร่

เครื่องของผมมันจะส่งผลลัพธ์เป็นตัวเลขนี้ออกมาครับ ...

```
8.7M    ./root
8.1M    ./boot
0       ./selinux
14M     ./sbin
8.0K    ./opt
4.1M    ./tmp
8.0K    ./initrd
248K    ./dev
16K     ./lost+found
251M    ./lib
5.4G    ./usr
53M     ./home
770M    ./proc
0       ./sys
8.0K    ./misc
16K     ./media
6.0M    ./bin
32K     ./windows
68M     ./etc
8.0K    ./.automount
8.0K    ./mnt
360M    ./var
8.0K    ./srv
6.9G    .
```

ดูแค่ไอ้ที่ป้ายเหลืองๆ เอาไว้นั่นก็แล้วกัน จะเห็นว่า /usr นั้นใหญ่โตมโหระทึกเลยทีเดียว เพราะผมเล่นติดตั้ง software ของ Fedora Linux ลงไปเกือบจะครบทุกตัว แล้วยังไปหามาเติมเข้าไปอีกตั้งเยอะ :-) ถ้าขืน follow ตามตัวเลขที่ผมลอกตำราเล่มอื่นมาตั้งแต่แรก ก็เป็นอันว่าติดตั้งไม่สำเร็จอยู่ดี ... เพราะฉะนั้นจึงไม่ต้องแปลกใจว่า ทำไมในอีกเครื่องหนึ่งนั้นผมแยก partition ของ /usr ออกมาตั้ง 7.9G เพราะว่าผมทุรนทุรายกับการติดตั้ง Linux บนเครื่องนู้นเครื่องนี้มาตั้งหลายหนแล้วนี่นา :-) สำหรับ partition อื่นๆ ก็เพี้ยนจากตำราไม่นักหรอกครับ /var อาจจะใหญ่ไปหน่อย แต่เพราะมีขยะเนื่องจากการ update แล้วยังไม่ได้ลบออกอยู่บ้างมันก็เลยดูตุงๆ พื้นที่ hard disk พอสมควร

ทีนี้ก็ได้รู้ขนาดที่ต้องการของตัวเองละ ... ถ้าอยากจะลงใหม่ด้วยการแบ่ง partition ให้เป็นระเบียบเรียบร้อยก็จำตัวเลขไปเผื่อๆ กันเอาเองก็แล้วกัน ถ้าจะบ้าเลือดกับ software ขนาดหนัก ก็ต้องเผื่อพื้นที่ของ /usr ไว้ให้มากๆ ... ลองไปดูซิว่า /usr มันไปโตๆ บวมๆ ตรงไหนมั่ง :-P

```
2.3G    ./share
22M     ./sbin
136M    ./include
88M     ./java
331M    ./local
2.1G    ./lib
120K    ./doc
1.6M    ./kerberos
2.5M    ./games
278M    ./bin
204M    ./X11R6
8.0K    ./etc
104K    ./src
31M     ./libexec
5.4G    .
```

ตัวที่กินพื้นที่ดุๆ เลยก็คือ /usr/share กับ /usr/lib ครับ ยังไม่ได้เข้าไปสำรวจเหมือนกันว่าอะไรจะต้องใหญ่โตขนาดนั้น? เพราะจริงๆ แล้วผมก็จับเอา fonts ของ MS-Windows ยัดเข้าไปใน directory ที่ชื่อว่า /usr/local/share/fonts/local ไปแล้ว ... เพราะงั้นที่ใหญ่บะลั่กกั๊กออกมานั่นมันจึงไม่ใช่ผลงานของผมโดยตรงอย่างแน่นอน :-)

จบเรื่องที่อยากจะเล่าในระดับประถม 1 แค่นี้ก่อนแล้วกันครับ ถ้าบังเอิญว่าใครอยากจะเริ่มติดตั้งใหม่อีกรอบหนึ่ง ก็จัดการกันซะตอนนี้เลย ... ผมขออนุญาตไปค้นตำราก่อนว่า ระดับประถม 2 นี่ Linux ควรจะต้องรู้คำสั่งอะไรที่มากกว่านี้มั่งดีกว่า :-D

หรือถ้าใครคิดว่าอยากจะทบทวนอีกครั้ง รายการคำสั่งข้างล่างนี้ก็คือที่ผมเล่าผ่านๆ มาทั้งหมดครับ หาอ่านรายละเอียดของแต่ละตัวกันเอาเองจากในเครื่องที่ทำการติดตั้งไว้ก็ได้

ที่ผมเรียกเป็นระดับอนุบาลมี ... ls, cp, rm, mv, mkdir, rmdir, cd, exit, logout และ man ... แล้วชุดต่อมาใน "*ประถมบท*" ผมเล่าเพิ่มเติมไว้แค่ 4 คำสั่งคือ chmod, su, df และ du

น่าจะเล่าคำสั่งไหนต่อไปอีกดีหว่า? ... เดี๋ยวค่อยว่ากัน!!

อ้อ !... เพื่อจะไม่เป็นการเสียเวลารอ ผมขอแนะนำให้ทดลองฝึก vi ไปเลยดีกว่ามั้ย ... ทำอย่างนี้ครับ เปิดเครื่องขึ้นมา login กันตรงหน้า graphic หรือจะออกไป login จาก virtual console ที่เป็น text mode ก็ได้ จากนั้นเรียกคำสั่ง **vimtutor** ขึ้นมา ... แค่นั้นแหละครับ ที่เหลือก็ทำตามตำราที่เขาเตรียมไว้ให้ไปเรื่อยๆ ... เท่านี้ก็พอจะใช้ vi ได้คล่องๆ ในระดับหนึ่งแล้วล่ะครับ... แฮ่ม !!

ประถมบทที่ 2 : อยู่ไหนหว่า?

อย่างที่เล่าเอาไว้แล้วว่า ตอนที่ติดตั้ง Linux โดยทั่วๆ ไปนั้น จะเกิดไฟล์ต่างๆ ในระบบของเราประมาณ 3 - 4 หมื่นไฟล์ด้วยกัน มีทั้งที่เป็นไฟล์คำสั่ง, บ้างก็เป็น config, บ้างก็เป็นคู่มือหรือ documents เพื่อให้เราศึกษาเพิ่มเติม, หรือไฟล์ประกอบที่จำเป็นบ้างไม่จำเป็นบ้างกับการทำงาน ขึ้นอยู่กับว่าเราจะบ้าเลือดขนาดไหนในตอนที่ติดตั้ง :-)

ทีนี้ ... พอใช้งานไปซักพักหนึ่ง เราก็จะเริ่มสับสนกับผลงานของตัวเอง เพราะเราซักไม่แน่ใจว่าไฟล์ที่เราต้องการแต่ละตัวนั้นมันอยู่ที่ไหน? หรือว่ามันยังมีอยู่ในระบบรึเปล่า? ในโลกของ MS-Windows ผมเคยเห็น user หลายคนใช้ระบบการค้นหาใน graphic mode หรือบางคนก็อาจจะใช้คำสั่ง attrib หรือแม้แต่ dir ที่มี /s เป็น parameter ตามหลัง ... ก็ถ้า Linux ไม่มี features เหล่านี้เลยก็ต้องถือว่าเชยละเอียดจริงๆ ... ถูกรึเปล่า?

คำสั่งที่ใช้ในการค้นหาไฟล์ของ Linux น่าจะมีอยู่ด้วยกันหลายวิธีครับ ขึ้นอยู่กับความชำนิชำนาญและตาม style การเรียกใช้คำสั่งของแต่ละคน เพราะคำสั่งของ Linux ได้รับการพัฒนามาจากหลายแหล่งทั่วโลก ... เราก็คงจะเล่ากันแค่กลุ่มคำสั่งมาตรฐานของมันก็พอ แล้วใครที่อยากจะลองวิธีอื่นๆ ก็หาโอกาสไปลองใช้ลองฝึกกันเอาเอง

คำสั่งในกลุ่มที่จะเล่าต่อไปนี้ก็จะมี locate หรือ slocate, whereis, which, และ find

slocate เป็นคำสั่งที่ปรับปรุงมาจาก locate โดยเพิ่มเติมรายละเอียดเกี่ยวกับด้าน security ลงไป แล้วก็เลยเรียกกันว่า slocate ... ปัจจุบันคำสั่ง locate ไม่มีอีกแล้วครับ แต่ที่พวกเรายังเรียกใช้มันได้ เป็นเพราะ Linux รุ่นหลังๆ จัดการให้คำสั่ง locate เป็น link ของ slocate ไปแล้ว เวลาเรียก locate หรือ slocate ก็เลยหมายถึงคำสั่งเดียวกัน ... วิธีใช้คำสั่งก็แค่ ...

locate <ชื่อไฟล์>

locate หรือ slocate นี้ทำงานได้ค่อนข้างเร็วครับ เนื่องจากมันจะมีการทำรายการ index หรือสารบัญของไฟล์ต่างๆ ในระบบเอาไว้ก่อน เรียกว่าพอหาปุ๊บก็เกือบจะเจอในทันทีเลย ... **ถ้าเรารู้ตัวสะกดที่แน่นอนของชื่อไฟล์นั้นๆ :-)**

แต่ถ้าบังเอิญไม่รู้จริงๆ หรือจำได้เพียงบางส่วน หรือไม่รู้ว่าไฟล์ที่ต้องการหานั้นถูกกำหนดให้มีชื่อเป็นตัวพิมพ์ใหญ่พิมพ์เล็กยังไงมั่ง ก็ต้องอาศัย parameter ช่วยนิดหน่อยครับ

locate -i <ชื่อไฟล์>

-i เป็น parameter ที่สั่งให้ locate ไม่ต้องสนใจ case ของตัวอักษรว่าจะเป็นพิมพ์เล็กหรือพิมพ์ใหญ่

locate -r <regular expression ของชื่อไฟล์>

ตัวอย่างเช่น ... locate -r grub*

แต่เพราะว่ามันจะทำงานกับ database ของระบบที่มีอยู่แล้วเท่านั้น ซึ่งเราก็อาจจะต้องคอย update ฐานข้อมูลนี้หรือทำการซ่อมสร้างตามแต่เวลาที่เห็นสมควร ซึ่งบางครั้งเราก็เลยอาจจะรู้สึกสะดวกกว่าถ้าจะใช้คำสั่ง find ทำหน้าที่นี้แทน

find ชื่อคำสั่งก็บอกไว้โจ่งแจ้งเลยครับ ผมเจอคำสั่งนี้ครั้งแรกด้วยการขอดูคู่มือมั่วๆ ว่า man find แล้วก็เลยเห็นวิธีการใช้ของมัน :-) ... เพราะฉะนั้นเวลานึกชื่อคำสั่งอะไรแล้วไม่แน่ใจว่าสะกดถูกรึเปล่า หรือไม่แน่ใจว่าจะเป็นคำสั่งที่ต้องการมั้ย ก็ให้ทดลองขอดูคู่มือจากการเรียกใช้ **man <ชื่อคำสั่ง>** ก่อนก็ได้ครับ

ส่วนใหญ่ของการใช้คำสั่ง find เป็นอย่างนี้ ...

find <starting path> -name <reg.exp ของชื่อไฟล์>

วิธีระบุชื่อไฟล์ที่ต้องการหาก็จะเหมือนกับคำสั่ง **locate -r** นั่นเอง คือสามารถใส่เครื่องหมาย * หรือ ? แทนตัวอักษรใดๆ ของชื่อไฟล์ได้

รายละเอียดและความสามารถของคำสั่ง find มีเยอะครับ เช่นหาว่าไฟล์ไหนถูกสร้างขึ้นมาหรือถูกแก้ไขเปลี่ยนแปลงในวันที่หรือเวลาไหน หรือจะหาไฟล์ที่มีขนาดเท่าไหร่ ... อย่างนี้เป็นต้น ... ก็ไปหาอ่านเอาเองจากการเรียก **man find** ก็แล้วกัน

whereis เป็นคำสั่งที่เอาหาไฟล์ประเภท binary, หรือ source code, หรือคู่มือการใช้ของคำสั่งใดคำสั่งหนึ่ง ... จะเห็นว่ามันมี scope แคบกว่า 2 คำสั่งแรกที่เล่าเอาไว้ในบทนี้ แต่ก็เหมาะกับการหาคำสั่งที่ต้องการโดยไม่ต้องยุ่งเกี่ยวกับชื่อไฟล์ที่ไม่เกี่ยวข้อง วิธีใช้ก็แค่ ... **whereis <ชื่อคำสั่ง>**

which ในทางหนึ่งมันทำหน้าที่คล้ายกับการหาด้วย **whereis** ต่างกันเพียงแค่ว่า **which** จะหยุดการค้นหาทันทีเมื่อมันพบคำสั่งนั้นๆ ใน path ที่ได้รับการกำหนดเอาไว้ ... ดังนั้นโดยลักษณะดังกล่าว เราจึงเอามันไว้เพื่อตรวจสอบว่าคำสั่งที่เราเรียกใช้อยู่เป็นประจำนั้น ใช่ตัวที่เราต้องการจริงๆ รึเปล่าด้วย เพราะระบบการค้นหาคำสั่งของ Linux ก็จะไล่ไปตามลำดับก่อนหลังของ path ที่กำหนดเอาไว้ก่อนแล้วตั้งแต่ตอนที่ login เข้าสู่ระบบของ user แต่ละคน

ครบแล้วนะ 4 คำสั่ง ปัญหาที่ยังจะตามมาอีกสำหรับ "*มนุษย์ขี้สงสัย*" ก็คือ เราก็หาจนเจอไฟล์ที่อยากจะหาแล้ว หรืออาจจะเห็นชื่อของไฟล์บางตัวที่น่าสนใจ ... แต่ในเมื่อ Linux ไม่ได้กำหนดเงื่อนไขของ executable files เอาจาก extension ของไฟล์นั้นๆ ... แล้วเราจะไปรู้ได้ยังไงล่ะว่ามันเป็นไฟล์ประเภทไหน? ... เราจะใช้คำสั่ง file ครับ :-) วิธีใช้ก็ง่ายนิดเดียว

file <ชื่อไฟล์>

ผลลัพธ์ที่ได้ก็จะแตกต่างกันไป เพราะบางตัวเป็น binary, บางตัวก็เป็น ASCII text file, บางตัวก็เป็น link, บางตัวก็เป็น directory, หรืออาจจะเป็น block device, ไม่งั้นก็เป็น shared object, ฯลฯ ไม่แน่ใจเหมือนกันว่ายังมีประเภทอื่นๆ อีกมั้ย :-) ... แต่ทั้งหมดก็จะมีแต่ ASCII text file เท่านั้นแหละครับที่เราสามารถอ่านออก แล้วก็ text file บางตัวมันก็เป็น script หรือ configuration ของ software ต่างๆ ในระบบของ Linux ด้วย ... แล้วก็เพราะ Linux ทำงานเกือบทุกอย่างอยู่บนพื้นฐานของ text file นี่แหละครับ ที่ทำให้ text edtor อย่าง **vi** หรือ **vim** รวมไปถึง **emacs** หรือ **joe** มีความสำคัญมากมายขนาดที่ user ทั่วๆ ไปนึกไม่ถึงกันเลยทีเดียว

การเปิดอ่าน text files ทั่วๆ ไปนั้นจะใช้คำสั่ง **cat**, **more**, หรือ **less** ... แต่บางคนก็ขี้เกียจจำกันหลายตัว เขาก็จะตะบันใช้ vi เพียงตัวเดียว เพื่อจัดการกับ text file ทุกตัวไปเลย ... เพียงแต่ในบางสถานการณ์นั้น การใช้ vi ซึ่งสามารถ edit แก้ไขเพิ่มเติมเข้าไปใน text file หนึ่งๆ ได้นั้น ก็ไปพัวพันกับเรื่องของ security หรือความ stable ของระบบเข้าให้เหมือนกัน ... มันคือที่มาของการสร้าง utilities เล็กๆ กลุ่มหนึ่งขึ้นมาเพื่อใช้ในการ "*อ่าน*" เพียงอย่างเดียว

คำสั่งทั้งหมดที่เล่ามานี้ ถือเป็นเรื่องพื้นฐานทั้งนั้นเลยครับ เพราะเรื่องราวของ Linux ต่อจากคำสั่งง่ายๆ เหล่านี้ก็คือการ setup หรือการแก้ไขเปลี่ยนแปลงค่า configuration ต่างๆ ของ software แต่ละตัว ... แน่นอนที่ Linux มีคำสั่งเป็นร้อยเป็นพันขนาดที่รวบรวมเป็นหนังสือได้หนาๆ ขนาดสมุดโทรศัพท์หน้าเหลืองเลยทีเดียว ซึ่งมีความจำเป็นต้องใช้ในสถานการณ์ที่ไม่เหมือนกัน ... ที่สำคัญก็คือหลายๆ คำสั่งก็เหมาะกับการเอาไปเขียนเป็น script มากกว่า เพราะมี syntax ที่ยุ่งยากทั้งยังสามารถ pipe (|) หรือ redirection (> และ >>) ให้ไปเกิดเป็นผลลัพธ์ต่างๆ ได้อีกสารพัดเรื่อง ... ก็คงถือโอกาสจบ "*ประถมบท*" กันแต่เพียงเท่านี้ และปล่อยให้ user ที่อยากจะเรียนรู้ได้ไปสัมผัสและฝึกหัดการใช้งานคำสั่งต่างๆ จนเกิดความชำนาญของแต่ละคนกันเอาเองต่อไป

อย่าลืมว่าคำสั่ง **man** คือคู่มือชิ้นสำคัญสำหรับการศึกษาค้นคว้าและทดลองใช้งาน Linux เพราะมันบรรจุแทบจะทุกคำสั่งเอาไว้ให้อยู่แล้ว และนอกจากนั้นก็อย่าลืมหัดใช้ **vi** หรือ **vim** จากเครื่องมือช่วยสอนที่แถมมาให้อย่าง **vimtutor** นั้นด้วย ... แม้ว่า power ของ Linux ส่วนหนึ่งจะอยู่ที่การกำหนดค่า configuration ต่างๆ ของระบบ แต่ power ที่แท้จริงในระบบของ Linux นั้นจะอยู่ที่การเขียน script ให้กับ shell ที่เราทำงานอยู่เป็นประจำมากกว่า ... หากใครที่ซุกซนพอจะเข้าไปซอกๆ แซกๆ กับแต่ละ directory ของ Linux แล้วก็จะเห็นว่า เราอาศัยการทำงานหรือการแก้ไข configuration ต่างๆ ของระบบผ่าน scripts นับร้อยๆ ที่คนอื่นๆ ช่วยกันเขียนเอาไว้ทั้งสิ้น :-)

ผมก็คงจะกระโดดข้ามไปเรื่องอื่นก่อนที่จะย้อนกลับมาที่เรื่องของ shell scripts เพราะรายละเอียดของการเขียน scripts นั้น น่าจะมีมากกว่าการกำหนดค่า configuration ของระบบซะอีก เพราะมันเกือบจะเป็นโลกของ developers จริงๆ ไปแล้ว ในขณะที่การติดตั้งระบบและการกำหนดค่า configuration นั้นยังเป็นโลกของ administrator เท่านั้นเอง

จบภาคแรก

การเขียนตำราไม่ใช่เรื่องง่าย ผมเองก็เขียนเล่าไปเท่าที่ตัวเองเคยรู้ และเอกสารฉบับนี้ก็อาจจะไม่ต่างไปจากบันทึกลำดับขั้นตอนของการเรียนรู้ Linux ของผมเอง ซึ่งอาจจะผิดเพี้ยนไปจากลำดับขั้นตอนของคนอื่นๆ อยู่บ้าง ... แต่นั่นไม่น่าจะใช่ปัญหา :-)

ผมเลือกใช้งาน Linux เพราะมัน "*ฟรี*" ... :-) นั่นอาจจะเป็นปัจจัยสำคัญประการแรกๆ ของหลายๆ องค์กร แต่ที่ผมชอบมันเพราะมันเปิดกว้างให้กับการเรียนรู้ แล้วมันก็ยังไม่จบทั้งที่มีความสมบูรณ์ในตัวของมันมากพอสมควรแล้ว ... มีเรื่องราวใหม่ๆ และมีสิ่งแปลกๆ ให้เราได้สัมผัสมันได้ตลอดเวลา ในขณะที่เราก็ยังสามารถใช้งานมันอย่างปกติทุกๆ วันต่อไปได้ ... ถ้าการเรียนรู้คือสิ่งที่สนุกสนานในชีวิตของพวกเรา ผมก็เชื่อว่า Linux คือทางเลือกที่น่าสนุกสนานนั้น

ผมเริ่มต้นจากสิ่งที่เล่ามาทั้งหมดนี้ แล้วก็ยังสนุกกับการอ่านและการเรียนรู้ features ใหม่ๆ ที่ยังซุกซ่อนอยู่ใน Linux ต่อไป มันอาจจะเป็นทั้งการปฏิบัติงาน และเกมล่าสมบัติสนุกๆ ในเวลาเดียวกัน ... มันเป็นสิ่งที่เราสามารถใช้ประโยชน์กับการปฏิบัติงานได้ และเป็นสิ่งเดียวกันกับที่เราจะสามารถมีสังคมเล็กๆ อีกสังคมหนึ่งที่จะพูดคุยแลกเปลี่ยนความคิดกันถึงเทคนิคหรือกลวิธีแปลกๆ บนเครื่องมือประจำโต๊ะหรือประจำตัวของพวกเราชิ้นที่เรียกว่าคอมพิวเตอร์นี้ ... มันอยู่ที่เราจะมองว่า Linux เป็นเรื่องที่น่า serious ที่ควรจะต้องจริงจังกับมันอย่างเอาเป็นเอาตาย หรือเราจะมองมันเป็น "*ของเล่น*" ที่สามารถใช้ประโยชน์กับการปฏิบัติงานประจำวันของเราได้ ?! ... เราก็ต้องเลือกของเราเองล่ะ ...

ผมจะเว้นวรรคทางเอกสารนี้ซักระยะหนึ่ง เพื่อทบทวนเรื่องที่เรียนรู้มาแล้วกับสิ่งที่กำลังเรียนรู้ใหม่ต่อไปเรื่อยๆ แล้วค่อยจับมาประมวลทั้งหมดเป็นเอกสารชุดที่ 2 ของการเผยแพร่ความรู้เกี่ยวกับ Linux ต่อไป ... บอกตรงๆ ว่ามันเขียนยากกว่าเอกสารบ้าๆ บอๆ ที่ตัวเองเคยเขียนไว้เยอะทีเดียว ... ก็สนุกของผมไปอีกแบบครับ :-)

Mr. Z.,
กรุงเทพฯ, 26.03.2005